REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mouloud MAMMERI de Tizi-Ouzou
Faculté de Génie Electrique et d'Informatique
Département d'Informatique





Mémoire de Fin d'étude
En vue de l'obtention du diplôme de Master en Informatique

Option : Conduite de Projet Informatique

Thème :

Conception et réalisation d'une application Client/Serveur pour la gestion de parc Informatique
Cas : Fonds National d'Investissement

Proposé & dirigé par :

Mr. S. HAMRIOUI

Mr.K.MAHFOUDI

Réalisé par :

Melle. AIT SEDDIK Hayat

Melle. BELLABIOD Meriem

Promotion 2011/2012

Remerciement

Nous remercions *DIEU* tout puissant de nous avoir donné la foi et le courage pour réaliser ce modeste travail et qui a mis dans notre chemin les bonnes personnes et nous a confié aux bonnes mains, à travers lui nous tenons à exprimer notre profonde gratitude à notre promoteur, Monsieur HAMRIOUI pour nous avoir encadré et guidé tout au long de notre projet, pour ses conseils judicieux et minutieusement prodigués.

Nos plus vifs remerciements vont aussi à tout le personnel du service Direction de l'Organisation et de l'Informatique du FNI, plus particulièrement, Monsieur MAHFOUDI qui nous a généreusement aidé durant notre stage, nous tenons aussi à leurs reconnaître le temps précieux qu'ils nous ont consacré au sein du Fonds National d'Investissement.

Que les membres du jury trouvent ici nos remerciements les plus distingués pour avoir accepté le jugement de notre travail.

Nos sincères sentiments vont à tous ceux qui, de près ou de loin, ont contribué à la réalisation de ce projet.

Dédicaces

Je dédie ce modeste travail à:

Mes très chers parents

Mes frères et ma sœur

Mes amis, en particulier

le Groupe G7

et à Tous ceux qui m'aiment

Hayat



Dédicaces

Je dédie ce modeste travail à:

Mes très chers parents

Mes adorables sœurs et mon frère

Mes amis, en particulier

le Groupe G7

et à Tous ceux qui m'aiment

Meriem



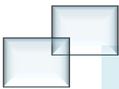
Tables des matières

Table des matières

Liste des figures

Liste des figures

Introduction



Introduction :

Rares sont maintenant les métiers qui n'ont pas de rapport avec l'informatique surtout avec la venue du troisième millénaire qui est caractérisé par une révolution technologique très importante et une croissance exponentielle de ces nouvelles technologies liées aux multimédias et aux télécommunications

La gestion et la maîtrise de l'information sont devenues des préoccupations de premier ordre pour toute organisation, notamment le Fonds National d'Investissement qui est doté d'un nombre conséquent d'outils informatiques de toutes sortes, qu'il faut connaître, suivre et maîtriser ces évolutions et dont il est difficile de gérer et de suivre le mouvement des équipements informatiques et de garder les traces de tous les incidents survenus.

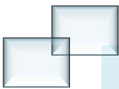
D'où le besoin d'un tel projet qui a pour but : inventorier le parc informatique avec ses machines (ordinateurs, imprimantes, composants) mais aussi les utilisateurs (personnes, services, lieux), les fournisseurs et les marques, ainsi que les interventions et le stock.

Notre travail consiste à concevoir et réaliser une application de gestion de matériel informatique pour le FNI tout en répondant au mieux aux besoins définis par l'organisme.

La gestion du parc informatique est le suivi en temps réel du patrimoine informatique, matériel et logiciel de l'entreprise. Elle offre une vision globale de l'état, du suivi et des coûts des appareils utilisés dans l'entreprise. Il s'agit non seulement de recenser les différents types de machines présentes dans l'entreprise, leur nombre, leur localisation mais de connaître aussi l'année d'acquisition de ces matériels, le nom des fournisseurs etc...

Avant de passer à l'implémentation de notre application il est évident de définir quelques concepts de bases, pour cela nous avons opté dans le premier chapitre à donner les différentes technologies informatiques qui nous renseignent sur les réseaux et l'architecture client/serveur, ensuite il est nécessaire de procéder à un diagnostic complet de l'organisme d'où nous allons présenter l'organisme FNI dans le deuxième chapitre, tandis que le troisième chapitre a été consacré au développement de cette solution, la démarche de l'élaboration de cette dernière s'appuie sur l'analyse et la conception à l'aide du langage de modélisation UML, les différents modules constituant notre système seront présentés. Le dernier chapitre est porté sur la réalisation et l'implémentation de la solution retenue.

Enfin, nous terminons par une conclusion générale.



Sans oublier les annexes qui fournissent quelques informations complétant le contenu de ce rapport. Nous trouverons dans l’annexe A les notions d’UML, les concepts de base de l’extension d’UML pour le web dans l’annexe B, tandis que l’annexe C contient des fiches utilisées par l’organisme.

Chapitre I:

Réseaux et architecture client/serveur

Introduction :

Un réseau est un ensemble de moyens matériels et logiciels, géographiquement dispersés, destinés à offrir un service comme les réseaux téléphoniques ou à assurer le transport des données. [1]

Les réseaux devront véhiculer toute sorte d'information avec des contraintes temporelles plus au moins fortes avec des volumes extrêmement variés.

Dans ce chapitre, nous allons aborder la notion de réseau et son architecture, puis le client/serveur, l'Internet, l'Intranet et leurs services.

I. Les réseaux :

I.1- Les réseaux Informatiques :

I.1.1- Définition d'un réseau informatique :

C'est une interconnexion d'ordinateurs, terminaux, serveurs, qui permettent un échange d'information entre ces derniers via un réseau de transmission. Cette connexion offre aux utilisateurs la possibilité de communiquer facilement.

Dans un premier temps, ces communications étaient destinées au transport de données informatiques, aujourd'hui les réseaux tendent à intégrer également la parole et la vidéo. [12]

I.1.2- Objectifs des réseaux :

La possibilité d'échange de données a permet :

- Un accès plus rapide aux informations numériques en se passant de supports de stockage lents.
- Une augmentation des capacités de stockage et leur structuration sur le réseau
- De bénéficier de manière distribuée des machines de traitement spécialisées: serveurs d'applications, meilleur partage des ressources actives.
- Multi diffusion rapide des messages.
- Accès à des banques de données collectives: *Internet – Intranet*

I.1.3- Les supports de transmission :

Le support de transmission est l'élément indispensable pour véhiculer l'information d'un émetteur vers un destinataire (récepteur).

Il existe deux catégories de support de transmission :

☞ Les supports physiques.

☞ Les supports hertziens.

I.1.3.1- Les supports physiques :

On distingue :

☞ **La paire de fils torsadés :**

C'est le support de transmission le plus simple et le moins couteux, composé de plusieurs paires de fils électriques, transmet les données à une vitesse de 100 Mbps sur une distance de 100 m, et facile à installer.

☞ **Le câble coaxial :**

Il est constitué de deux conducteurs de même axe séparés par un isolant permettant de le protéger des perturbations externes.

☞ **La fibre optique :**

Moins encombrante, et très utilisée. Elle est constituée de fibres de verre qui véhiculent l'information sous forme de signaux lumineux, elle est caractérisée par une large bande passante qui permet de transférer une grande quantité d'informations, et assure une minimisation du taux d'erreurs de transmission car elle est moins sensible aux interférences électromagnétiques grâce à son revêtement en plastique.

I.1.3.2- Les supports hertziens :

☞ **Les supports Hertziens par satellite :**

Basés sur des satellites permettant l'émission et la réception de données à partir de terminaux de petite dimension.

☞ **Les supports Hertziens par ondes radio :**

Ils utilisent la technologie WLL (Wireless Local Loop), le raccordement entre un abonné et la station centrale s'effectue par l'intermédiaire de fréquences radio.

I.1.4- Classification des réseaux informatiques :

La classification des réseaux se distingue en deux :

I.1.4.1- Selon l'envergure :

• **les réseaux personnels ou PAN** (Personal Area Network) **:**

Désignent un réseau restreint d'équipements informatiques habituellement utilisé dans le cadre d'une utilisation personnelle. Les bus les plus courants utilisés sont l'USB, les technologies sans fil telles que Bluetooth ou IR (infra rouge). [13]

• **Les réseaux locaux ou LAN** (Local Area Network) **:**

Sont des réseaux informatiques à une échelle géographique relativement restreinte, par exemple une salle informatique, une habitation particulière, un bâtiment ou un site d'entreprise. [13]

• **Les réseaux métropolitains ou MAN** (Metropolitan Area Network) **:**

Désignent des réseaux étendus d'ordinateurs habituellement utilisés dans les campus ou dans les villes. Ces réseaux utilisent généralement des fibres optiques. Par exemple, une université peut avoir un MAN qui lie plusieurs réseaux locaux. [13]

• **Les réseaux étendus ou WAN** (Wide Area Network) **:**

Sont des réseaux informatiques couvrant une grande zone géographique, typiquement à l'échelle d'un pays, d'un continent, voire de la planète entière. Le plus grand WAN est le réseau Internet. [13]

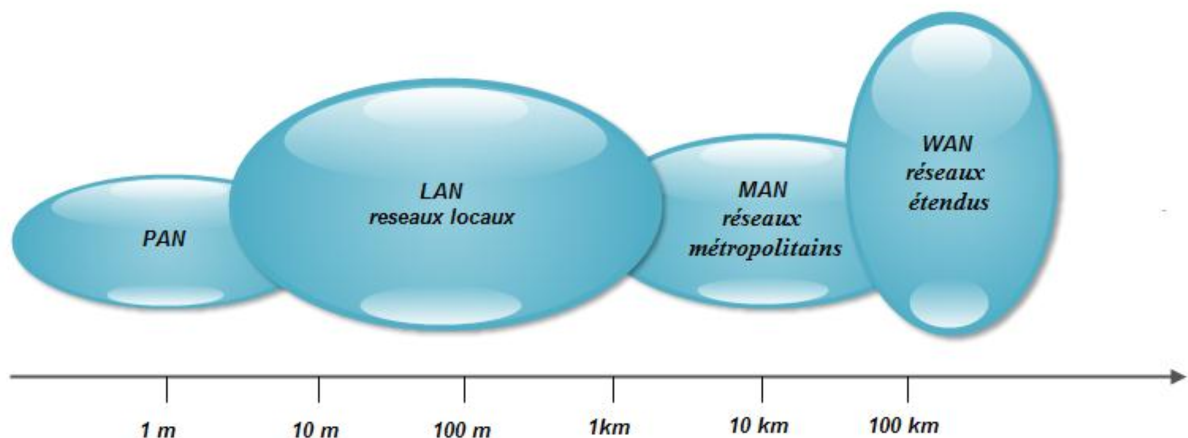


Figure I.1 : Classification des réseaux informatiques selon l'envergure

I.1.4.2- Selon la topologie :

On peut différencier les réseaux selon leur topologie, on trouve :

a. en bus :

Cette topologie consiste à relier chaque ordinateur à un bus par l'intermédiaire de câbles coaxiaux, l'information envoyée par un poste est diffusée en même temps vers tous les utilisateurs mais seul le poste destinataire est censé le prendre en compte. [13]

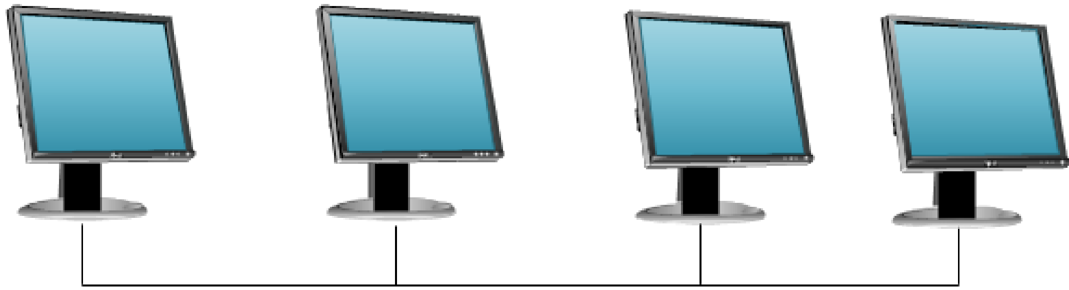
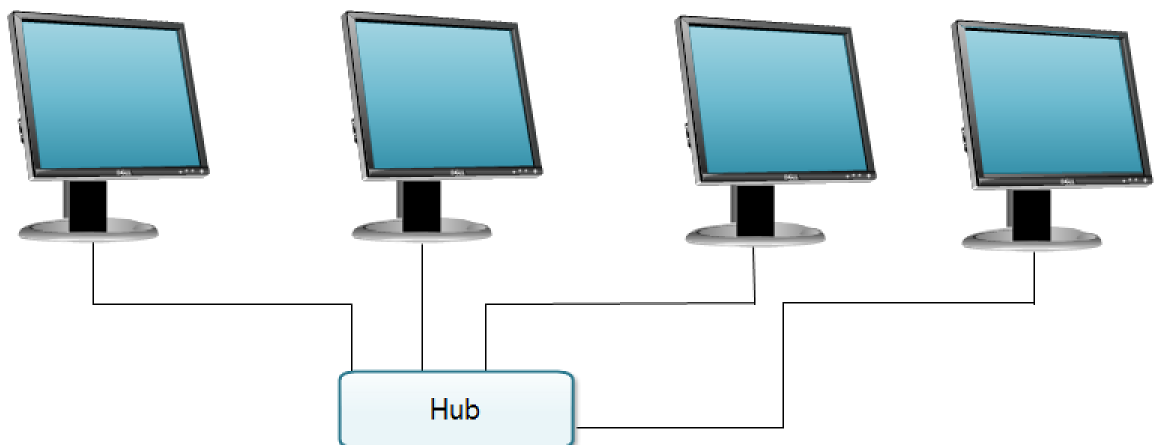


Figure I.2 : topologie en bus

b. en étoile :

Les stations du réseau communiquent bidirectionnellement avec le serveur. C'est une liaison point à point entre le serveur (hub ou Switch) et chaque machine. Deux stations peuvent échanger des données à condition d'en passer par le serveur. [13]



FigureI.3 : Topologie en étoile

c. en anneau :

Les stations sont connectées par un câble formant un anneau (ring). Dans cette topologie le message est transmis d'une station à une autre jusqu'à destination. Chaque station recevant un message vérifie s'il lui est destiné si c'est le cas elle le garde sinon elle le transmet à la station voisine. [13]

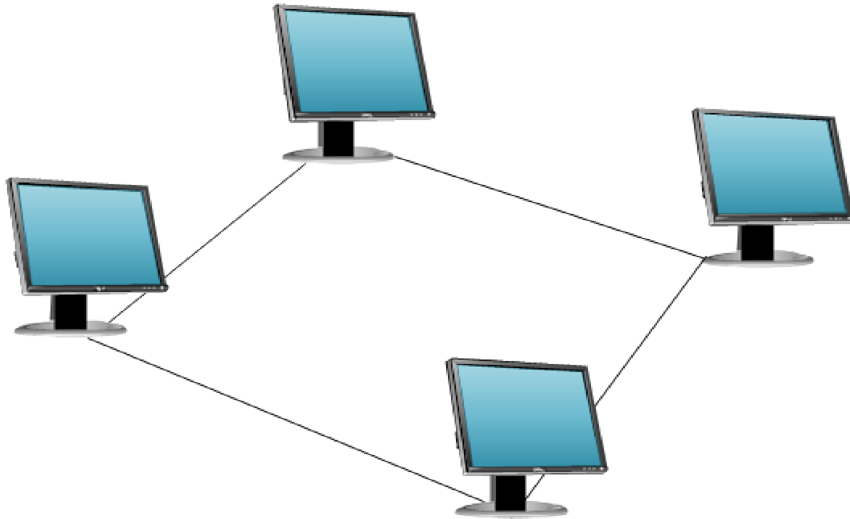


Figure I.4: Topologie en anneau

d. réseau maillé :

Un réseau maillé est caractérisé par le fait que deux nœuds quelconques soient reliés l'un à l'autre. Ce type de réseau permet plus de souplesse et de fiabilité dans son utilisation, mais il est difficilement envisageable pour un grand nombre de nœuds. Généralement, le maillage n'est pas parfait, on dit que c'est un réseau partiellement maillé. Tous les autres types de réseaux sont des sous-ensembles de ce type, obtenus en posant des conditions sur les liens entre les nœuds. [13]

Remarque :

Toute topologie adoptée doit faire au préalable l'objet d'une étude prenant en compte plusieurs facteurs : Nombre de stations à connecter, flux des données, Coût, distance entre entités communicantes, évolution possible, Résistance aux pannes et lignes de secours, administration …etc.

I.1.5- Modes de transmission de données dans un réseau informatique :

Le but de la transmission de données est de les acheminer d'une source vers une destination d'une façon synchrone ou asynchrone, en parallèle ou en série.

I.1.5.1- Transmission parallèle et transmission série :

- **Transmission parallèle :**

Les bits d'un même caractère sont envoyés sur des fils distincts pour arriver simultanément à destination. Elle est rapide mais très couteuse et très encombrante surtout sur les longues distances.

- **Transmission série :**

Contrairement à la transmission en parallèle, les bits sont envoyés l'un derrière l'autre sur un même fil de transmission.

I.1.5.2- Transmission synchrone et transmission asynchrone :

- **Transmission synchrone :**

Les caractères sont envoyés de manière irrégulière, l'intervalle de temps entre deux caractères est aléatoire, ils sont accompagnés de deux signaux START et STOP permettant de spécifier le début et la fin de chaque caractère.

- **Transmission asynchrone :**

Un fil particulier reliant la source à la destination transporte le signal d'horloge et les bits sont transmis l'un à la suite de l'autre à chaque top d'horloge.

I.1.6- Modes d'exploitation :

Il existe plusieurs types d'exploitation des canaux de transmission :

I.1.6.1- Le mode simplex :

On parle de transmission unidirectionnelle si une seule extrémité peut émettre.

I.1.6.2- Le mode semi duplex :

Appelé aussi bidirectionnel à l'alternat, permet une transmission dans les deux sens mais pas en même temps.

I.1.6.3- Le mode duplex :

C'est du bidirectionnel simultané, permet une transmission dans les deux sens simultanément.

I.1.7- Architecture des réseaux :

Le transport des données d'une extrémité à l'autre d'un réseau nécessite un support physique de communication. Cependant, pour que ces données arrivent correctement au destinataire, avec la qualité de service exigée, il faut une architecture logicielle. L'ensemble de protocoles nécessaires constitue une architecture. Un protocole est l'ensemble de règles qui définissent les modalités de fonctionnement d'une communication entre deux ordinateurs. [9][10]

I.1.7.1- Architecture OSI (Open System Interconnexion) :

Que l'on appelle Open System Interconnexion ou Interconnexion des Systèmes Ouverts, c'est une architecture qui forme le modèle de référence qui comporte sept couches. Cette architecture est illustrée par la **figure I.5 [4]**

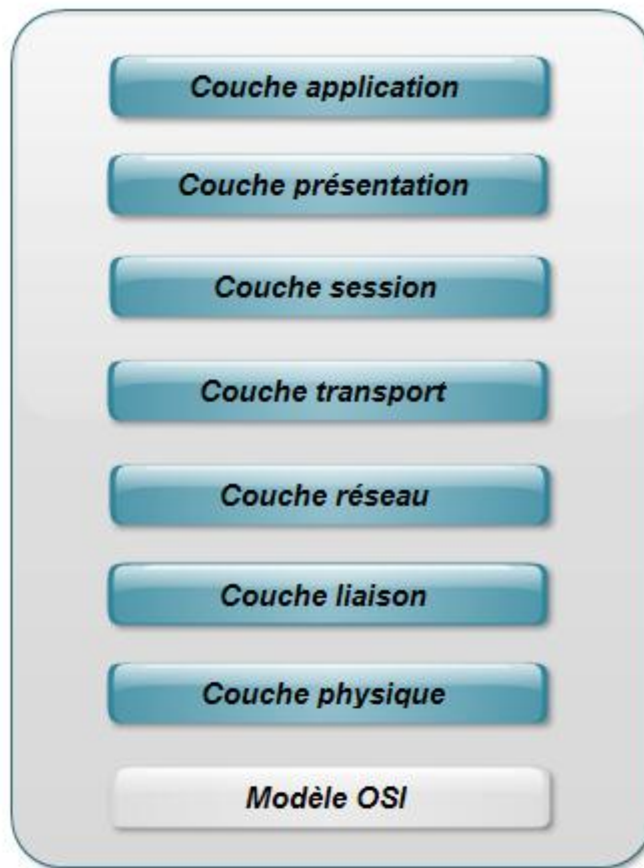


Figure I.5: Les couches du modèle OSI.

1. La couche physique :

Elle est responsable de la manipulation des détails mécaniques et électriques de la transmission physique d'un flot de bits sur le canal de communication.

2. La couche de liaison :

Sa tâche principale est de détecter et de corriger les erreurs de transmission, l'émetteur traite également les accusés de réception émis par le récepteur.

3. La couche réseau :

Elle s'occupe du routage et du contrôle de congestion. L'information stockée au niveau de cette couche est structurée sous forme de paquets de données.

4. La couche transport :

Elle est responsable du transfert de données entre les entités de session. Ce transport devant être transparent, c'est-à-dire indépendant de la succession des caractères et même des éléments binaires transportés.

5. La couche session :

Elle permet à des utilisateurs de différentes machines d'établir des sessions entre eux.

Parmi ses services : la gestion du dialogue, le transfert de fichiers, …

6. La couche présentation :

Elle gère la syntaxe et la sémantique des informations transmises (le codage, la compression, la cryptographie des données, …).

7. La couche application :

Elle est responsable de l'interaction directe avec les utilisateurs. Elle traite les protocoles de connexion à distance, le courrier électronique ainsi que la définition des terminaux virtuels.

I.1.7.2- L'architecture TCP/IP (Transmission Control Protocol / Internet Protocol) :

Dans les années 70, DOD (Department Of Defense), devant le foisonnement de machines utilisant des protocoles de communication différents et incompatibles, décide de définir sa propre architecture. L'architecture TCP/IP est à la source du réseau Internet de nombreux réseaux privés. Cette architecture comme celle d'OSI est répartie en couches **(figure I.4)**.

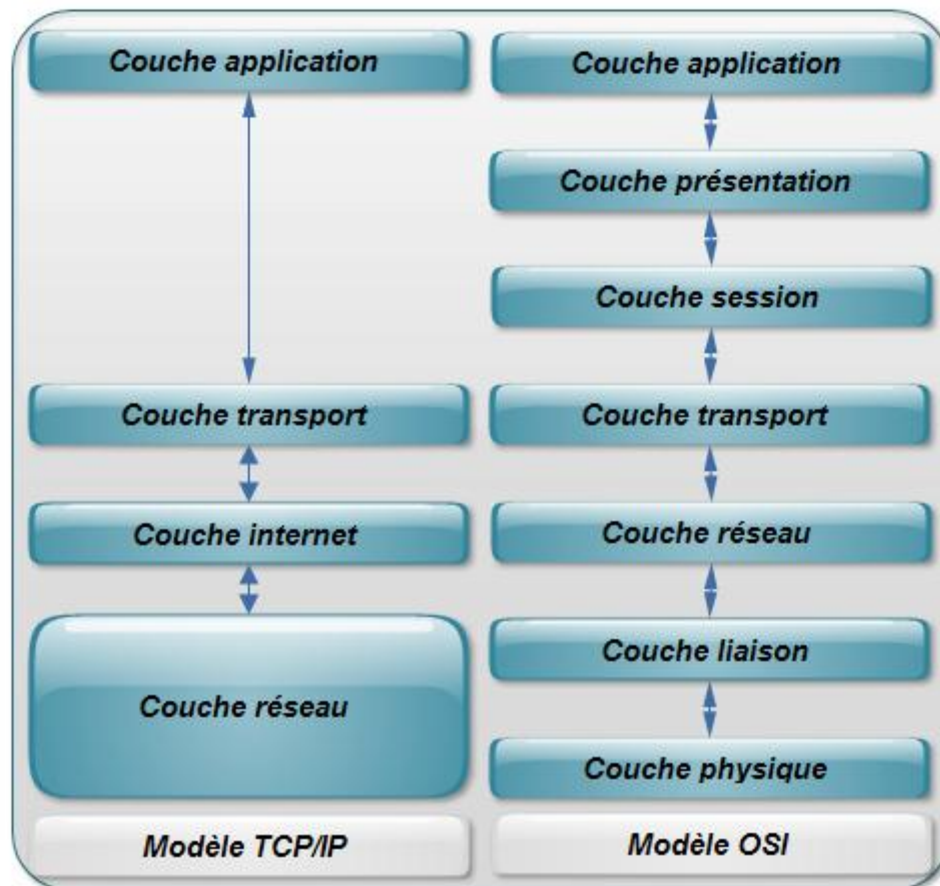


Figure I.6 : La pile de protocole TCP/IP avec sa correspondance OSI.

- **La couche application :**

Elle englobe les applications standard du réseau :

SMTP (Simple Mail Transport Protocol) : permet d'échanger du courrier entre deux serveurs de messagerie.

TELNET : Protocole permettant de se connecter sur une machine distante (serveur) en tant que utilisateur.

FTP (File Transfert Protocol) : Est un protocole permettant d'échanger des fichiers via Internet.

DNS (Domain Name Service) : Service de nom du domaine. Il permet de convertir le nom d'une machine en une adresse réseau (ou IP) et vice versa.

- **La couche transport :**

Elle assure l'acheminement des données sur le réseau local et les mécanismes permettant de connaitre l'état de la transmission. Les protocoles de cette couche permettent d'envoyer des informations d'une machine à une autre.

TCP : Il assure le contrôle de données, orienté connexion.

UDP (User Datagram Protocol) : Archaïque et non orienté connexion, il n'assure aucun contrôle de transmission de données.

- **Couche Internet :**

La couche Internet est chargée de fournir le paquet de données. Elle contient cinq protocoles, les trois premiers sont les plus importants :

IP : gère les destinations des messages, adresse du destinataire.

ARP (Address Resolution Protocol) : gère les adresses des cartes réseau. Chaque carte a sa propre adresse d'identification codée sur 48 bits.

ICMP (Internet Control Message Protocol) : gère les informations relatives aux erreurs de transmission. Il ne corrige pas les erreurs, mais signale aux autres couches que le message contient des erreurs.

RARP (Reverse Address Resolution Protocol) : gère l'adresse IP pour les équipements, qui ne peuvent s'en procurer une, par lecture d'information dans un fichier de configuration. En effet, lorsque le PC démarre, la configuration réseau lit l'adresse IP qu'elle va utiliser. Ceci n'est pas possible dans certains équipements qui ne possèdent pas de disques durs (imprimantes, terminaux).

IGMP (Internet Group Management Protocol) : permet d'envoyer le même message à des machines faisant parti d'un groupe. Ce protocole permet également à ces machines de s'abonner ou de se désabonner d'un groupe. Ceci est utilisé par exemple dans la

vidéo conférence à plusieurs machines, envoi de vidéos,…

- les réseaux de mobiles :

La première génération est apparue à la fin des années 70, le réseau est composé de cellules, c'est-à-dire de zones géographiques limitées à quelques kilomètres qui recouvrent le territoire de l'opérateur. Ces cellules se superposent partiellement pour assurer une couverture complète du territoire cible. Le mobile communique par une interface radio, avec l'antenne qui joue le rôle d'émetteur/récepteur de la cellule.

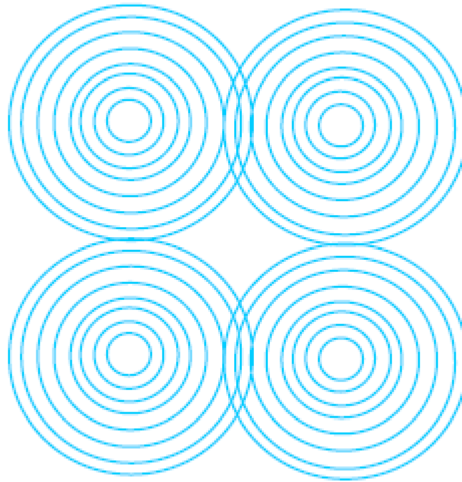


Figure I.7 : Les cellules des réseaux mobiles

I.2- L'architecture client/serveur :

De nombreuses applications fonctionnent selon un environnement client/serveur, cela signifie que des *machines clientes* (des machines faisant partie du réseau) contactent un *serveur*, une machine généralement très puissante en terme de capacité d'entrée-sortie, qui leur fournit des *services*. [7]

L'architecture client/serveur a deux composants principaux :

- **Un composant client** : c'est une machine qui fournit à l'utilisateur toute la gamme de ses services pour exécuter des applications.
- **Un composant serveur** : c'est une machine généralement très puissante en terme de capacité d'entrée/sortie, qui fournit aux clients des services de gestion de données, de partage d'information, d'administration du réseau et de sécurité.

I.2.1- Fonctionnement d'un système client/serveur :

Un système client/serveur fonctionne selon le schéma suivant :

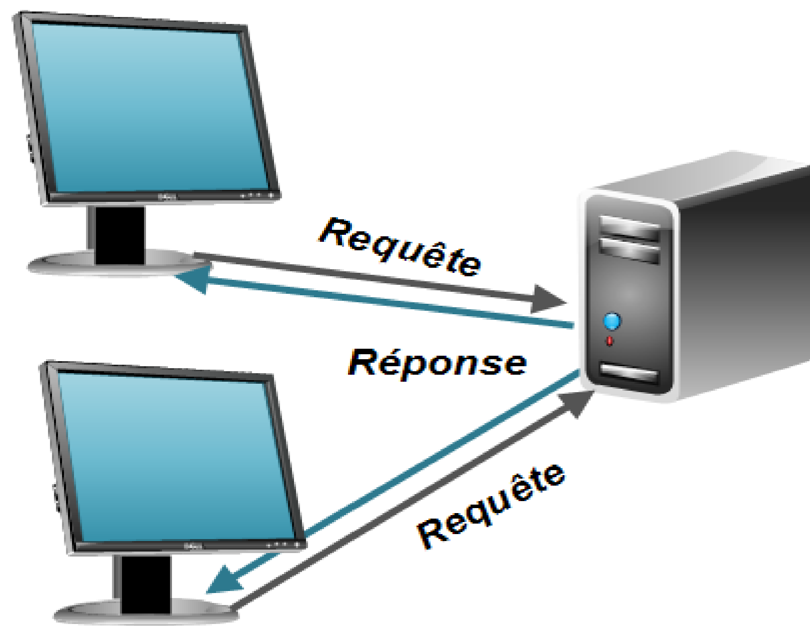


Figure I.8 : Système client/serveur.

1. Le client émet une requête vers le serveur grâce à son adresse IP et le port, qui désigne un service particulier du serveur. [06]

2. Le serveur reçoit la demande et répond à l'aide de l'adresse de la machine client et son port. [O6]

I.2.2- Classification des architectures clients/serveurs :

I.2.2.1- Architecture à 2 niveaux :

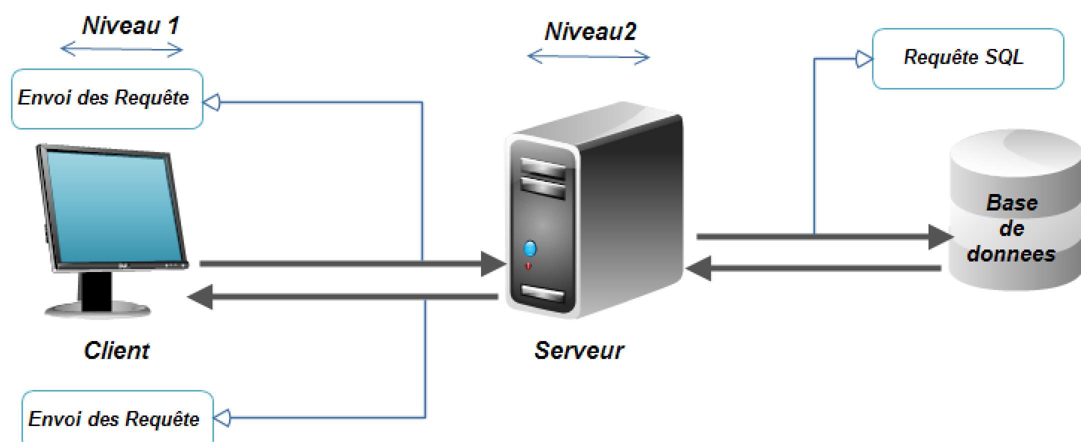


Figure I.9: Architecture client/serveur à 2 niveaux.

L'architecture à deux niveaux (aussi appelée *architecture 2-tiers*, *tier* signifiant *étage* en anglais) caractérise les systèmes clients/serveurs dans lesquels le client demande une ressource et le serveur la lui fournit directement. Cela signifie que le serveur ne fait pas appel à une autre application afin de fournir le service. [15]

Avantages :

- **Des ressources centralisées** : Etant donné que le serveur est au cœur de l'infrastructure, il peut gérer des ressources communes à tous les clients, comme par exemple une base de donnée centralisée afin d'éviter les problèmes de redondance et d'incohérence.
- **Une meilleure sécurité** : le nombre de points d'accès aux données étant moins important.
- **Un réseau évolutif :** on peut supprimer et/ou rajouter des clients sans perturber le fonctionnement du réseau et sans faire des grandes modifications.

Inconvénients :

- Dépendance totale avec l'architecture des réseaux et les systèmes d'exploitation.
  Une modification de l'application ou de la structure de la base de données nécessite un redéploiement sur les postes clients.
- Coûts élevés, nécessite des serveurs très performants et une connexion réseau à haut débit.

I.2.2.2- Architecture à 3 niveaux :

⇒ Dans l'architecture à 3 niveaux (appelée *architecture 3-tiers*), il existe un niveau intermédiaire, c'est-à-dire que l'on a généralement une architecture partagée entre :

⇒ Un **client** : le demandeur de ressources, équipé d'une interface utilisateur (généralement un navigateur Web) chargé de la présentation. [06]

⇒ Le **serveur d'applications** (appelé également **middleware**) : chargé de fournir la ressource mais faisant appel à un autre serveur.

⇒ Le **serveur de données** : fournissant au serveur d'application les données dont il a besoin.

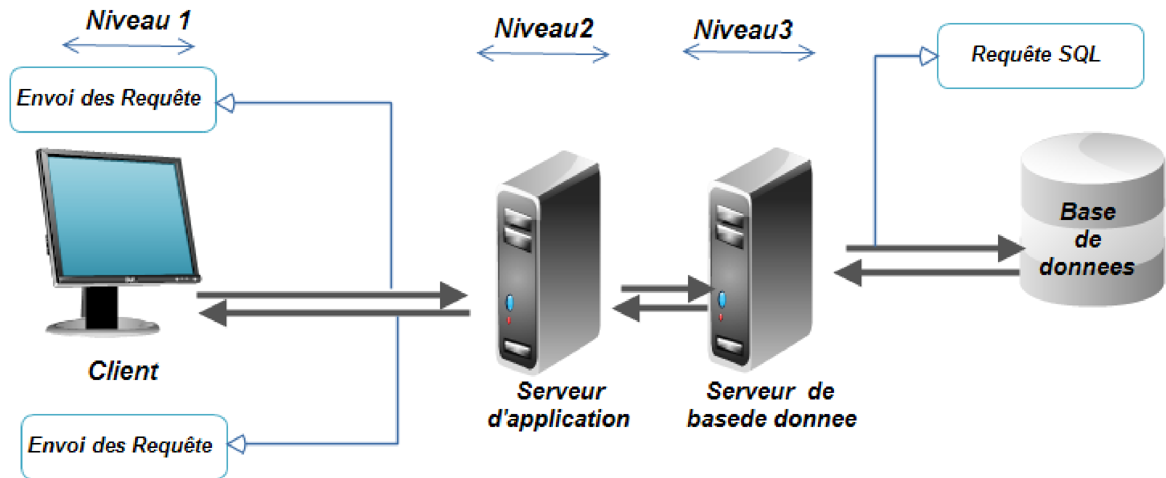


Figure I.10 : Architecture client/serveur à 3 niveaux.

En plus des avantages d'une architecture client/serveur, cette infrastructure procure :

- L'administration est centralisée au niveau serveur pour tous les clients ayant accès.
- Une plus grande sécurité : elle est définie pour chaque service.
- Permet une configuration matérielle et logicielle hétérogène des postes clients.
- Tendance au couplage faible : possibilité de remplacer un composant par un autre
- Passage à l'échelle, gestion d'utilisateurs concurrents.

Inconvénients :

- Ticket d'entrée technologique plus élevé
- Applications transactionnelles parfois complexes à développer.

I.2.2.3- Architecture multi-niveaux :

Dans l'architecture à 3 niveaux, chaque serveur (niveaux 2 et 3) effectue une tâche (un service) spécialisée. Un serveur peut donc utiliser les services d'un ou plusieurs autres serveurs afin de fournir son propre service. Par conséquent, l'architecture à trois niveaux est potentiellement une architecture à N niveaux. [06]

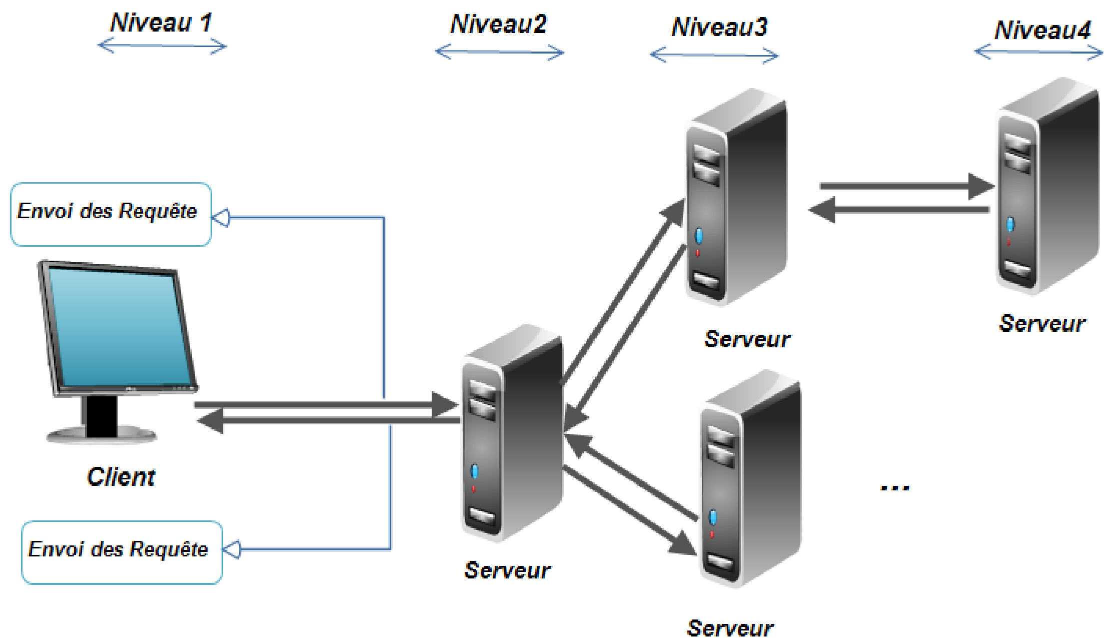


Figure I.11 : Architecture client/serveur à N niveaux.

I.2.3- Les apports et les contraintes de l'architecture client/serveur :

Dans ce qui suit nous allons voir concrètement quels sont les avantages de l'architecture client/serveur par rapport aux systèmes classiques et quelles sont ses limites.

a) Les apports :

- **Capacité de traitement :** le modèle client/serveur s'appuie sur les progrès réalisés dans le domaine des microprocesseurs. Les techniques qu'il met en œuvre nécessitent des capacités de traitement importantes que les microprocesseurs d'aujourd'hui sont capables de fournir.
- **Capacité de mémorisation :** dans une architecture client/serveur, les postes serveurs doivent gérer simultanément les contextes applicatifs de plusieurs postes clients. Cette architecture sera d'autant mieux adaptée aux environnements transactionnels lourds où les serveurs disposent de grandes capacités mémoire.
- **Standardisation :** en s'appuyant sur les normes standards, le modèle client/serveur assure la portabilité des applications entre systèmes distincts, et garantit leur inter-portabilité.
- **Productivité du développeur :** l'environnement client/serveur facilite le processus de développement applicatif. Cela est dû notamment à la séparation fonctionnelle qui constitue le principe de base du modèle client/serveur.

- **Portabilité et interopérabilité :** le modèle client/serveur assure le partage d'une même application sur différentes plates-formes, échanger des informations et faire interagir plusieurs applications dans des environnements hétérogènes.

b) Les contraintes

- **complexité de mise en œuvre :** une complexité qui se manifeste notamment lors du découpage des traitements entre partie cliente et partie serveur.
- **Fragilité relative du système :** dans un environnement centralisé, les données sont concentrées en un seul endroit, le contexte client/serveur peut nécessiter l'installation des données sur des serveurs distincts. Cette répartition géographique des données et l'utilisation même d'un réseau fragilise l'ensemble du système et pose alors le problème de la sécurité des données en environnement réparti.
- **Investissements initiaux élevés** : la mise en place d'une architecture client/serveur nécessite des efforts financiers importants, investissement en matériels et logiciels tels que micro, interfaces graphiques, matériels de connexion.

I.2.4- Comparaison entre l'architecture à deux niveaux et l'architecture à trois niveaux :

L'architecture Client-serveur à deux niveaux ne présente aucun majeur inconvénient lorsqu'il s'agit de l'utiliser pour un système à un seul site, bien au contraire elle offre des avantages concernant la simplicité et la facilité de développement. Mais lorsque le client/serveur s'est agrandi pour traiter des applications critiques d'ordres intergalactiques, l'architecture Client-Serveur à trois niveaux s'est révélée essentielle car moins coûteuse. [11]

Le tableau suivant représente la comparaison des deux architectures :

Architecture à deux niveaux	Architecture à trois niveaux
-l'application tourne sur le client et toute modification doit être opérée sur tous les clients d'où la complexité d'administration du système. -sécurité faible. -de nombreuses instructions SQL transitent le réseau ce qui engendre une faible performance. -le changement d'échelle est difficile. -Réutilisation faible (les programmes sont monolithiques).	-l'application est gérée de manière centralisée d'où l'administration du système est moins complexe. -sécurité importante. -réduction du trafic grâce aux procédures implantées sur le serveur d'où une bonne performance. -possibilité de la répartition de la charge sur plusieurs serveurs ce qui facilite le changement d'échelle. -réutilisation importante (les programmes sont constitués de plusieurs composants).
-facilité de développement.	-difficulté de développement.
-le support de l'Internet est faible (bande passante limitée).	-excellent support de l'Internet.
-ne supporte pas les bases de données hétérogènes.	-supporte les bases de données hétérogènes.
-la souplesse de l'architecture matérielle est très limitée.	-la souplesse de l'architecture matérielle est excellente.

On peut constater que l'utilisation d'une architecture 3-tiers offre beaucoup plus d'avantages, de flexibilité et de souplesse d'utilisation qu'une architecture 2-tiers.

L'architecture que nous mettrons en œuvre sera donc de type 3 niveaux pour les raisons suivantes :

- la nature de notre Outil (ouvert et extensible) ;
- les utilisateurs de cet outil et les circonstances d'utilisations.

Conclusion :

Au cours de ce chapitre, nous avons présenté certains concepts des nouvelles technologies de l'information et de la communication, à citer : les réseaux informatiques, l'internet et le web, l'architecture client/serveur…

Les différents concepts traités dans ce chapitre nous aiderons à mieux comprendre notre mode d'opération et les notions fondamentales pour mener à bien notre application.

Chapitre II:

Organisme d'accueil

Introduction :

Pour bien cerner notre sujet d'étude, il est primordial de comprendre la structure et le fonctionnement de notre structure d'accueil afin de répondre aux besoins de notre étude.

A cet effet, ce chapitre est consacré pour objectif principal de donner une vue globale de la société Fonds National d'Investissement, sa création, ses missions, sa structure…ensuite prendre connaissance du domaine d'étude dans lequel la société souhaite améliorer son fonctionnement.

II. Organisme d'accueil :

II.1- Présentation du FNI :

En vue de promouvoir de nouveaux instruments à l'intervention de l'Etat dans le financement du développement, les pouvoirs publics ont décidé de la création d'un fonds National d'Investissement dans le cadre du parachèvement du processus de réforme du secteur financier et bancaire engagé par l'Etat.

II.1.1- Création du FNI :

L'actuel organisme Fonds National d'Investissement « FNI » a été sous la dénomination de la Banque Algérienne de Développement « BAD » (ex Caisse Algérienne de Développement) créée par une loi du 7 Mai 1963.

Elle a eu pour mission fondamentale concourir au financement des investissements productifs, et à la mise en œuvre des plans et programmes d'investissement, en vue de la réalisation des objectifs de développement économique du pays. [16]

Elle répondait alors, à la nécessité de disposer d'une institution spécialisée dans l'octroi de crédits, à moyen et long terme, en faveur du secteur public ou privé.

II.1.1- Missions du FNI :

Les principales missions dévolues au FNI, consistent notamment en :

II.1.2.1- FINANCEMENT D'OPERATIONS DU SECTEUR ECONOMIQUE :

- Financement de projets d'investissement économique.
- Partenariat avec les entreprises publiques et/ou des investisseurs étrangers pour la

réalisation de projets de développement.

- Promotion du financement bancaire local nécessaire aux grands projets.
- Octroi de garanties.

L'intervention du FNI se fait sous la forme de prêts directs, de cofinancement avec les Banques ou par le biais de prise de participations.

L'un des objectifs du FNI est de mettre à la disposition des Entreprises des financements à durée longue et à des conditions avantageuses leur permettant de réduire la charge des financements des projets et accroitre leur rentabilité.

Pour cela, en plus de ses fonds propres, le FNI pourra mobiliser des ressources additionnelles via des prêts et avances du Trésor public ou recourir au marché obligataire avec ou sans garantie de l'Etat. [17]

II.1.2.2- FINANCEMENT DES PROJETS D'EQUIPEMENT PUBLIC :

- Le FNI agit en qualité de comptable assignataire et contrôleur financier.

Ces projets inscrits au budget de l'Etat sont financés par des concours définitifs ou sont couverts par des financements mixtes (budget/trésor).

II.1.3- ORIENTATIONS STRATÉGIQUES :

Le FNI a pour objectifs:

- le financement du développement de l'économie nationale
- la promotion des IDE.
- la promotion du financement bancaire local des grands projets.

Le FNI accompagne:

- les projets de toute nature décidés par l'Etat
- les projets économiques du secteur public et/ou du secteur privé dont l'activité participerait :
- à la substitution aux importations de biens et services;
- à la croissance des exportations hors hydrocarbures;
- au financement d'équipements publics par le marché, en substitution au budget de l'Etat.

Le FNI intervient par :

- des prises de participations,
- des financements directs,

- l'octroi de garanties.

L'octroi de garanties

Le Fonds octroie des garanties sur crédits extérieurs, à la demande d'opérateurs économiques algériens, en faveur de banques et institutions financières étrangères qui leur ont octroyé des prêts.

◊ Taux de commission 1% l'an de l'encours de crédit.

◊ Echéances semestrielles

Le Fonds octroie des garanties commerciales, au bénéfice d'opérateurs nationaux, sur ordre de banques de fournisseurs étrangers de biens et services dans le cadre de la réalisation de projets en Algérie. Elles comprennent :

◊ Les garanties de soumission dans le cadre d'appels d'offres couvrant la défaillance d'un soumissionnaire.

◊ Les garanties de restitution d'avance sur approvisionnement ou sur travaux.

◊ Les garanties de bonne exécution.

Les garanties sont délivrées au **taux de 1% l'an** (0.25% par trimestre indivisible)

Frais : contrevaleur de 2500 DA. [17]

II.1.4- L'organigramme de FNI :

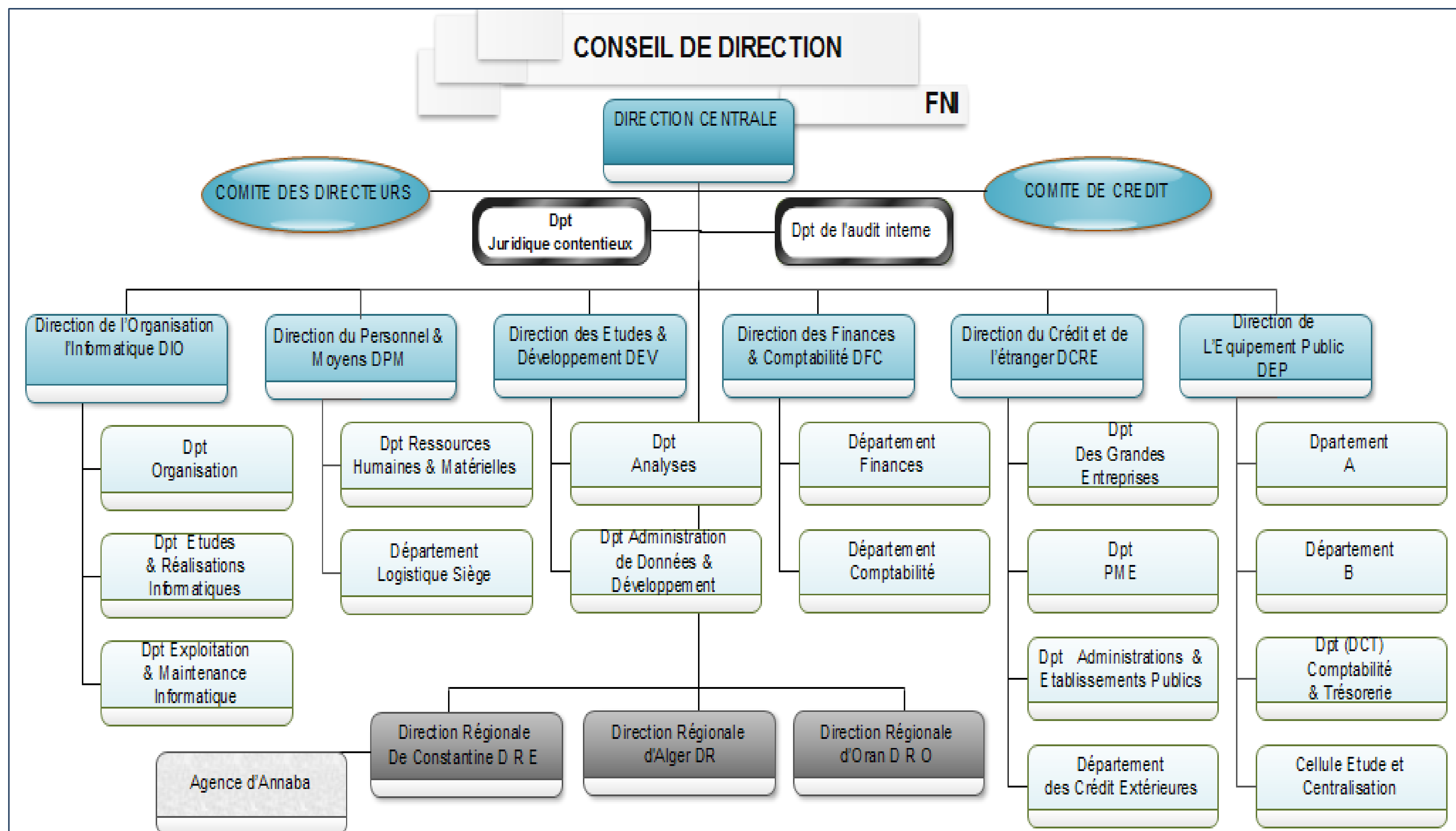


Figure II.1 : Organigramme de l'organisme de FN

II.1.5- Structuration des Sociétés :

L'organisation avec les missions de chaque département du Fonds National d'Investissement est organisée de la manière suivante:

☞ Comité de direction:

- ✓ présidé par le PDG de l'institution,
- ✓ constitué des Directeurs de l'ensemble des structures centrales et régionales.
- ✓ chargé du suivi de l'activité et du bon fonctionnement des structures
- ✓ s'assure notamment de l'élaboration, du suivi de l'exécution après leur adoption par le conseil de Direction, des programmes d'action ainsi que de leur évaluation.
- ✓ coordonne le traitement des affaires qui font intervenir plusieurs structures de la Banque. [16]

☞ Département de l'audit interne :

- ✓ vérification et évaluation des activités et opérations de l'institution relatives aux opérations bancaires et financières, à son organisation et sa gestion administrative
- ✓ veiller au respect des procédures
- ✓ veiller au suivi de l'exécution des décisions de la Direction générale et des autorités financières
- ✓ veiller à l'optimisation des procédures, la qualité des informations financières et leur conformité à la réglementation
- ✓ procéder aux contrôles inopinés, ainsi que toutes missions d'inspection
- ✓ procéder aux diagnostics et aux évaluations de l'ensemble des activités de la Banque,
- ✓ veiller à la protection, la sauvegarde du patrimoine de la Banque et d'amélioration de ses performances.
- ✓ veiller â l'application des procédures internes.
- ✓ évaluer l'efficacité des procédures de la Banque et proposer des recommandations nécessaires. [16]

☞ Département juridique et contentieux :

- Gestion des affaires en contentieux relevant du portefeuille
- Gestion des affaires en contentieux relevant des aspects administratifs
- Assistance juridique aux structures [16]

☞ **Département de crédit et de l'étranger:**

- ✓ Participer à la définition de la politique de Crédit de la Banque :
- ✓ en élaborant et proposant la politique générale de crédit au comité de crédit de la banque
- ✓ mettre en œuvre la stratégie de la banque en matière de crédits internes et externes
- ✓ évaluer et gérer les risques de crédit [16]

☞ **Direction du département public :**

- ✓ gérer pour le compte du trésor les opérations d'équipements publics inscrites au budget de l'état et bénéficiant d'un financement extérieur en gestion BAD.

  A ce titre:
- ✓ elle assure la mission de contrôleur financier pour les opérations dont elle a la charge
- ✓ elle assure le rôle de comptable assignataire et tient ses comptes conformément à la comptabilité publique
- ✓ elle procède aux différents mandatements sollicités par les ordonnateurs au profit de leurs cocontractants
- ✓ appels de fonds et Gestion des ressources budgétaires
- ✓ exécution des paiements
- ✓ Comptabilité des opérations d'Equipements publics
- ✓ Elle gère les crédits de paiement mis à la disposition des ordonnateurs par les organismes compétents [16]

☞ **Direction régional (ALGER, ORAN, CONSTANTINE) & agence ANNABA :**

- ✓ la mise en œuvre des directives instruites par la Direction Générale
- ✓ la réalisation des objectifs de la Banque
- ✓ la gestion du Portefeuille qui lui est affecté
- ✓ la gestion, le suivi et le recouvrement des créances de la Banque
- ✓ procéder à l'évaluation et à la supervision des projets
- ✓ la gestion et le suivi des rapprochements et assainissement des comptes financiers
- ✓ élaboration, consolidation du budget et suivi des réalisations budgétaires
- ✓ rationaliser la gestion des moyens humains et matériels de la structure
- ✓ veiller à la protection et à la sauvegarde du patrimoine [16]

☞ **Direction des études et du développement :**

- ✓ assurer et développer la fonction « Etudes », afin d'asseoir une ingénierie en matière d'analyse financière, de sélection et d'évaluation des projets et d'entreprises
- ✓ élaborer le plan de développement de la Banque
- ✓ mettre en place et gérer une banque de données économiques et financières au service de l'institution
- ✓ la Législation et la réglementation en vigueur du statut de la banque, dans le cadre de la politique générale arrêtée par le Conseil de Direction de la Banque, des orientations et des instructions de la Direction Générale de la Banque. [16]

☞ **Direction des finances et de comptabilité :**

- ✓ contribuer à l'élaboration de la politique de développement de la banque et de sa mise en œuvre par une meilleure adéquation des moyens financiers nécessaires à sa réalisation sous forme d'objectifs.
- ✓ s'assurer de l'optimisation du niveau de richesse de la Banque, par la mise en place d'un système de mesure, d'appréciation et de contrôle devant permettre la stabilisation et l'augmentation progressive du bénéfice net de la banque
- ✓ élaborer tout état financier et comptable
- ✓ assurer la fonction comptable de la banque
- ✓ entretenir des relations fonctionnelles avec le système contrôle interne et externe pour ce qui est de son activité. [16]

☞ **Direction du personnel et des moyens :**

- ✓ organiser et contrôler les moyens et le personnel nécessaires à l'activité de la banque. A ce titre, elle joue le rôle du conseil auprès de l'ensemble des entités afin de leur permettre d'optimiser le rendement et l'efficacité du personnel dont elles ont la responsabilité.
- ✓ organiser la gestion des biens meubles et immeubles de la banque et assurer l'entretien des biens situés au siège
- ✓ conseiller la Direction Générale en matière de gestion des relations sociales.
- ✓ mettre en œuvre et assurer le suivi d'une politique cohérente et dynamique de gestion des ressources humaines et matérielles de la banque [16]

Dans cette première partie nous avons présenté l'organisme d'accueil, la deuxième partie sera consacrée à l'étude de l'existant.

II.2- Présentation du domaine d'étude :

Le service informatique se trouve au niveau de la Direction de l'Organisation et de l'Informatique au 9ième étage de la société. En effet, c'est ce même Service Informatique qui gère l'ensemble des ressources informatiques de l'ensemble des départements des directions.

Il s'occupe de l'administration dans le domaine informatique par la mise en œuvre d'une politique d'organisation de la banque et notamment :

- ✓ participer à l'élaboration des plans d'organisation de la banque
- ✓ participer à la définition et la mise en place du système d'information de la banque
- ✓ assister les structures dans la prise en charge des problèmes d'organisations et procédures

Gérer les moyens informatiques de la banque et notamment :

- ✓ élaborer et mettre en œuvre des plans directeurs informatiques
- ✓ exploiter et maintenir les équipements informatiques et annexes
- ✓ exploiter et maintenir les applications et données informatiques
- ✓ réaliser les études et développements informatiques

Assister en permanence les structures à l'utilisation des outils informatiques [16]

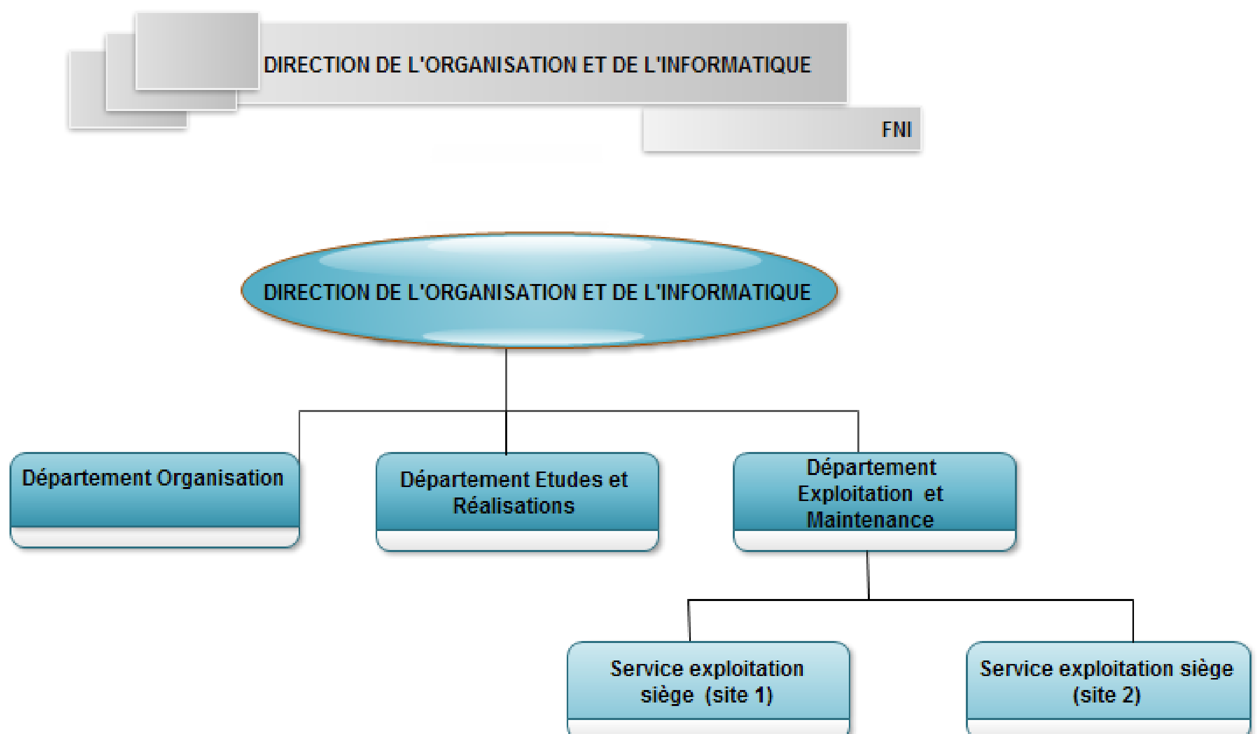


Figure II.2: Organigramme de la Direction de l'Organisation et de l'Informatique

II.3- Description du thème et problématique :

Il.3.1- Description du thème :

La mise en place d'une application informatique pour assurer une gestion des problèmes informatiques de l’ensemble des directions est une demande de la Direction de l’Organisation et de l’Informatique (DOI) pour répondre aux trois objectifs suivants:

- ✓ avoir une meilleure maîtrise des problèmes informatiques sur l'ensemble des directions;
- ✓ une célérité dans la résolution des incidents informatiques;
- ✓ un inventaire pour le matériel informatique et les incidents survenus ;
- ✓ une meilleure gestion du patrimoine informatique de l’organisme.

D'où le thème que nous avons à traiter**:**

« - Gestion des incidents du parc informatique. »

Il.3.2- Problématique **:**

L'ensemble des départements du FNI dispose d'un important lot de matériels informatiques qui sont gérés par un même service, la DOI. A l'heure actuelle, il n'existe pas de mécanisme fiable pour assurer un suivi du mouvement des équipements informatiques et du traitement

des incidents. Il est également difficile pour le service informatique de garder une trace des incidents qui surviennent.

Toutes ces difficultés sont liées au fait que la gestion est toujours manuelle.

Il nous revient alors de proposer une solution informatique à même de résoudre les difficultés que connaît le service informatique dans la gestion des incidents et du parc informatique.

II.4- Suggestions :

- L’automatisation de certains traitements liés à la gestion du parc informatique
- Mettre à la disposition des utilisateurs un logiciel qui leur permet de :
  - gérer les mouvements liés au traitement des incidents informatiques survenus ;
  - faire le suivi et l’inventaire du matériel informatique.

II.5- Solution **:**

Suite à notre analyse de la situation actuelle, dans cette partie nous allons développer une application qui permettra d'automatiser certains traitements liés à la gestion du parc informatique et ce logiciel se résume à :

Implémenter une application client-serveur 3 tiers qui sera intégrée au système réseau, où la base de données sera installée sur un serveur et sera accessible par des postes clients (futurs utilisateurs de notre application) et ainsi répondra aux objectifs suivants :

- ⇒ faire le point du parc informatique à tout moment;
- ⇒ identifier de manière unique chaque équipement du parc;
- ⇒ localiser de façon précise chacun de ces équipements;
- ⇒ suivre tous les incidents matériels et logiciels;
- ⇒ organiser les interventions aux incidents selon la priorité des besoins;
- ⇒ faire les statistiques et les inventaires des différents incidents;
- ⇒ mémoriser la description de chaque incident.

Conclusion :

Nous avons tenté, tout au long de cette partie, de décrire fidèlement la situation actuelle de notre organisme d’accueil. Cette description nous a aidé d’une part à bien comprendre le fonctionnement général de cet organisme et d’autre part à relever les dysfonctionnements qui seraient à l’origine des difficultés que rencontre l’organisme.

En somme, suite à notre analyse de la situation actuelle, nous avons proposé une solution informatique que nous avons jugée mieux adaptable aux objectifs définis précédemment.

Celle-ci ne va pas se réaliser sans la mise en place d’un système d’information qui répondra aux besoins des futurs utilisateurs. Cet aspect fera l’objet de la partie ultérieure.

Chapitre III:

Analyse et conception

Introduction :

Avant toute réalisation d'une application informatique, il convient de suivre une démarche méthodologique et rigoureuse pour planifier et concevoir l'application, en mettant en évidence tous les objectifs tracés pour la bonne élaboration du projet souhaité.

Dans ce chapitre, nous allons présenter le cycle de développement suivi qui est l'Orienté Objet. Ensuite, nous ferons une analyse détaillée qui mettra en évidence les différents acteurs intervenant dans l'application. Enfin, en s'appuyant sur les résultats de la phase analyse, la phase conception donnera la modélisation des objectifs à atteindre.

Pour ce faire, on va s'appuyer sur le langage de modélisation UML (Unified Modeling Language).

III- Analyse & Conception :

III.1- Approche de modélisation :

La modélisation de notre application suivra l'approche orientée objet vu la nature dynamique et complexe du problème étudié.

III.1.1- le choix de l'approche orientée objet :

L'approche orientée objets considère un système comme une collection d'objets interdépendants qui communiquent entre eux par l'intermédiaire de messages et concourrent à l'exécution de tâche. Le choix de cette approche résulte de la stabilité de la modélisation par rapport au monde réel qui est en changement perpétuel. En outre, elle offre une meilleure fiabilité provenant de la réutilisation des éléments d'un développement à un autre, un développement plus rapide grâce au prototypage et la réutilisation, une meilleure adéquation des systèmes modélisés ou réalisés aux besoins des utilisateurs grâce à sa grande richesse sémantique qui œuvre à une communication efficace entre utilisateur et développeur, mais également une plus grande flexibilité des systèmes réalisés et des coûts de maintenance réduits, parce que des objets peuvent être modifiés sans conséquence pour les autres objets.

En effet, notre système se veut être un système ouvert, multifonctionnel et dynamique. L'interdépendance de l'approche orientée objets, nous permettra d'intégrer facilement de nouvelles contraintes sous forme de nouveaux objets s'appuyant sur un tel outil, peut être aussi étayé par le fait que notre système sera intégré dans un environnement où baignent d'autres systèmes avec lesquels il pourra collaborer.

III.1.2- Présentation de l'UML :

UML (Unified Modeling Language), que l'on peut traduire par « langage de modélisation unifié » est une notation permettant de modéliser un problème de façon standard. Ce langage est né de la fusion de plusieurs méthodes existantes auparavant, et est devenu désormais la référence en terme de modélisation objet, à un tel point que sa connaissance est souvent nécessaire pour obtenir un poste de développeur objet.

Le méta-modèle UML fournit une panoplie d'outils permettant de représenter l'ensemble des éléments du monde objet (classes, objets, ...) ainsi que les liens qui les relient. Toutefois, étant donné qu'une seule représentation est trop subjective, UML fournit un moyen astucieux permettant de représenter diverses projections d'une même représentation grâce aux **vues** qui sont constitués d'un ou plusieurs diagrammes.

III.2- Démarche de modélisation de notre application :

Pour ne pas mélanger les problèmes, notre démarche de conception est découpée suivant les trois points de vue classiques de modélisation, en insistant pour chacun sur le ou les diagrammes UML prépondérants

Le langage UML propose 9 diagrammes répartis sur les trois axes du niveau conceptuel :

- Fonctionnel
- Structurel
- Temporel

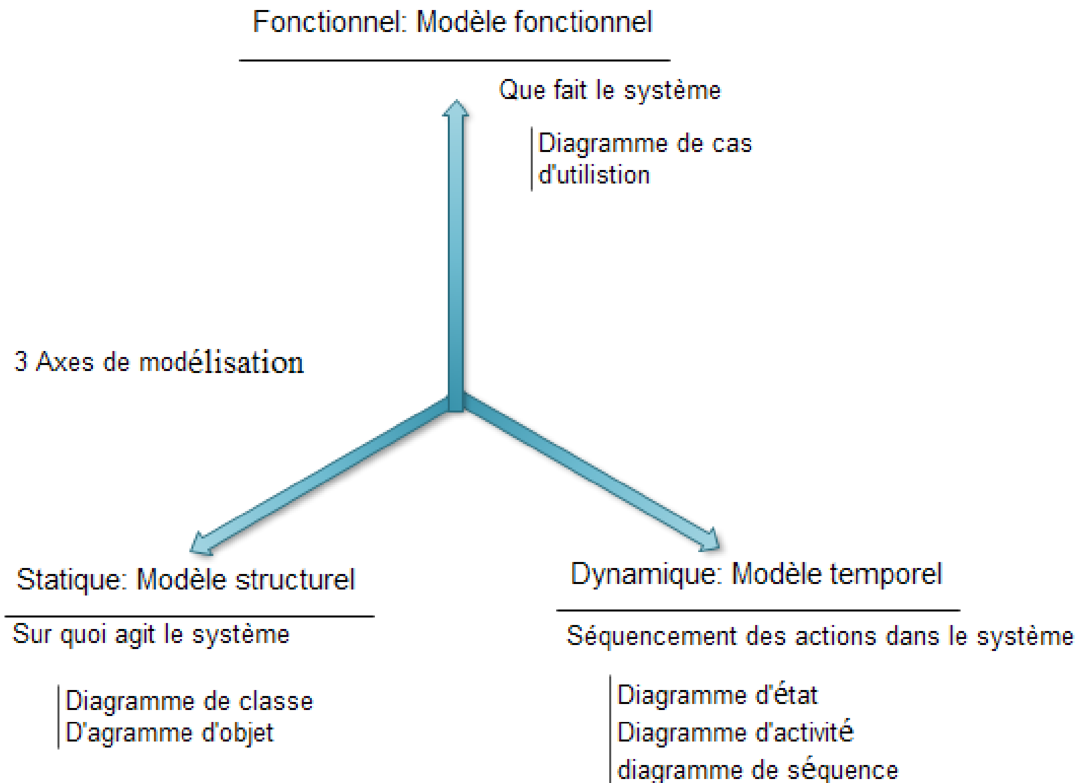


Figure III.1: Les trois composants du langage UML.

Toutes les vues proposées par UML sont complémentaires, elles permettent de mettre en évidence les différents aspects du logiciel à réaliser. On peut organiser une présentation d'UML autour d'un découpage en vues, ou bien différents diagrammes, selon qu'on sépare plutôt les aspects fonctionnels des aspects architecturaux, ou les aspects statiques des aspects dynamiques. [18]

III.2.1- La vue fonctionnelle :

Est représentée à l'aide de diagramme de cas d'utilisation et de séquences. Elle cherche à appréhender les interactions entre les différents acteurs/utilisateurs et le système, sous forme d'objectifs à atteindre d'un côté et sous forme de séquence d'interactions de l'autre.[18]

III.2.2- La vue structurelle, ou statique :

Réunit les diagrammes de classes et les diagrammes de packages. Les premier favorisent la structuration des données et tentent d'identifier les objets composants constituant le programme, les attributs, opération et méthodes, ainsi que les liens et les associations qui les unissent. Les second s'attache à regrouper les classes fortement liées entres elles en des composants les plus autonomes possibles. A l'intérieur de chaque package, on trouve un diagramme de classe. [18]

III.2.3- La vue dynamique :

Qui est exprimée par le diagramme d'état. Cette vue est plus algorithmique et orientée « traitements », elle vise à décrire l'évolution (la dynamique) des objets complexes du programme tout au long de leur cycle de vie. De leur naissance à leur mort, les objets voient leur changement d'état guidé par les interactions avec les autres objets.

Dans notre conception, on a retenu le diagramme des cas d'utilisation pour présenter l'aspect fonctionnel de notre système, le diagramme de séquence pour illustrer son aspect dynamique et le diagramme de classe pour présenter la structure de l'application. [18]

III.3- Les diagrammes UML utilisés :

Pour la conception de notre projet, on a utilisé les diagrammes suivants :

- **Diagramme de cas d'utilisation :** appelés également les Use Cases, ils permettent d'une part une meilleure délimitation du système et d'autre part, l'identification des acteurs du domaine et les actions qu'ils souhaitent effectuer à travers l'application.
- **Diagramme de séquence:** Fortement utilisé pour réaliser le passage entre les cas d'utilisation à la modélisation d'objets et diagrammes de classe. Il permet la représentation des interactions entre objets selon un point de vue temporel.
- **Diagramme d'activité :** Décrit le comportement d'une méthode, le déroulement d'un cas d'utilisation et les enchainements d'activités. Ils sont adaptés pour modéliser la logique de contrôle et montrent aisément l'organisation séquentielle globale des activités de plusieurs objets et cas d'utilisation, ce qui est profitable à notre cas d'étude.
- **Diagramme de classe :** Ils décrivent les types des objets qui composent un système et les différents types de relations statiques qui existent entre eux. Les diagrammes de classes font abstraction du comportement du système.
- **Diagramme de déploiement :** l'utilité de ce digramme découle du fait qu'il permet la représentation de l'environnement de développement de notre application.

Une description plus détaillé du formalisme et des concepts du langage UML est présenté dans l'annexe A.

La figure suivante donne la représentation graphique de la démarche de modélisation choisie pour la conception de notre application :

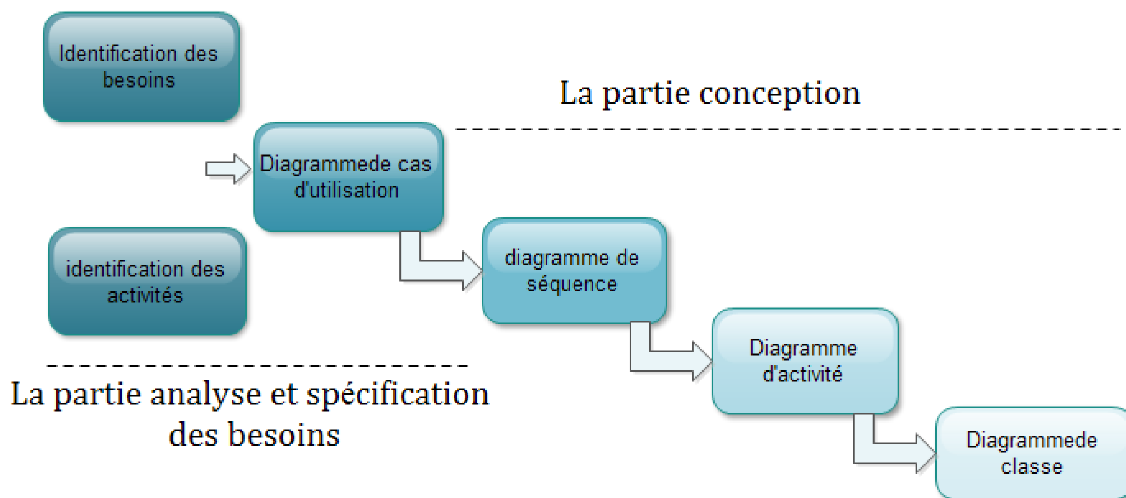


Figure III.2 : La démarche de modélisation de l'application

UML n'est pas une méthode objet dans la mesure où elle ne présente aucune démarche. A ce titre UML est un formalisme de modélisation objet. Le mot méthode parfois utilisé par abus de langage ne doit donc pas être entendu comme une démarche de développement. Pour cette raison nous allons suivre le cycle de développement orienté objet

III.4- Cycle de développement orienté objet :

Pour garantir nos chances de réussite nous allons suivre une méthode de développement qui permet de mieux organiser, d'avoir une meilleure compréhension, de réduire la complexité des applications et de faciliter l'interprétation des concepts logiciels, pour y aboutir nous allons suivre les étapes de développement orienté objet qui sont : analyse, conception, réalisation et maintenance :

III.4.1- Analyse :

Elle offre une description complète, cohérente, lisible et révisable du problème à partir de laquelle on pourra commencer à concevoir une solution.

L'analyse orientée objet comprend le processus d'identification et de modélisation des classe d'objets essentielles et les relations et interactions entre elles.

III.4.2- Conception :

Elle permet de fournir un ensemble de modèles traduisant la solution pour le problème donné. La conception orientée objet repose sur la notion d'objet pour établir ces modèles.

III.4.3- Réalisation :

Elle permet d'obtenir la base de données et le logiciel correspondant à notre conception.

III.4.4- Maintenance :

Durant l'exploitation du produit logiciel, les développeurs seront sollicités pour ajouter de nouvelles fonctionnalités ou codifier des comportements existants.

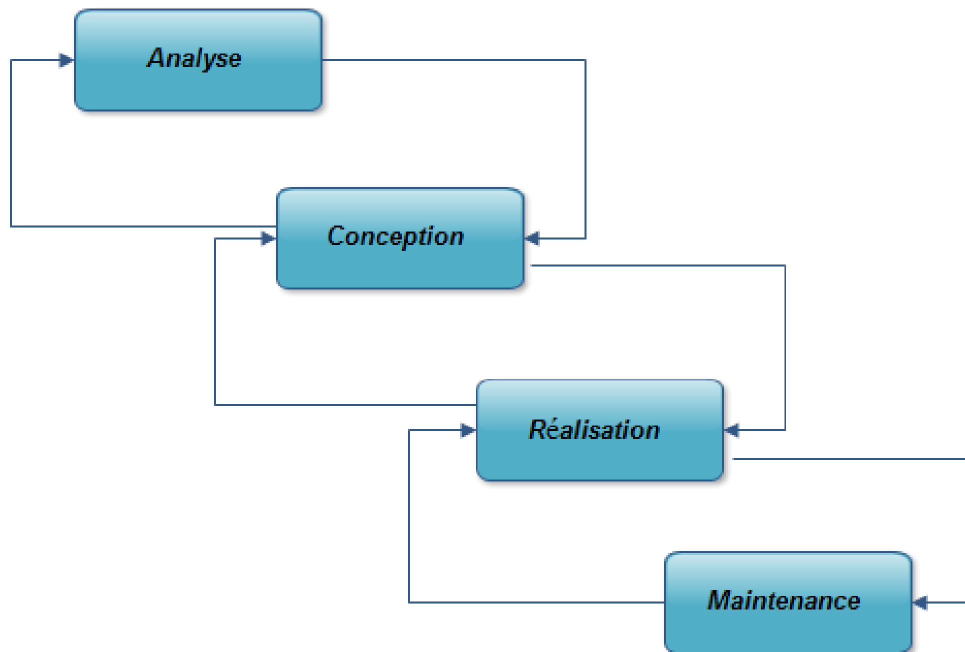


Figure III.3 : Cycle de développement orienté objet.

III.5- Etapes d'élaboration de notre projet :

Le processus de développement de notre application est scindé selon les étapes suivantes :

Partie Analyse :

À partir de la définition des besoins, nous allons identifier les acteurs et les tâches desquels nous déduirons les cas d'utilisation.

Partie conception :

Établir un ensemble de scénarios d'utilisation standards à l'aide des diagrammes de séquences et d'activités pour modéliser les activités d'une opération spécifique.

Ces diagrammes nous aiderons à identifier les classes nécessaires, nous permettant d'aboutir au diagramme de classes.

III.5.1- Analyse :

Cette partie répondra à la question suivante : que fait le système ? , pour y parvenir, nous allons procéder en premier lieu au listing des résultats attendus par l'application, en terme de fonctionnalités, performance, maintenance, sécurité, etc. (besoin non fonctionnels). En second lieu, nous allons mettre en évidence les différents acteurs intervenants dans le système, ainsi que leurs besoins exprimés dans les différentes interviews en utilisant les diagrammes de cas d'utilisation.

III.5.1.1- Définition des besoins :

Notre projet porte sur la réalisation d'une application client-serveur 3-tiers pour l'automatisation des procédures de gestion de tout le matériel informatique ainsi que les utilisateurs au niveau des différents départements de la société. L'application assurera un environnement interactif afin de rendre les tâches de gestion accessibles aux différents acteurs, et ce via un réseau.

III.5.1.1. 1- Besoins non fonctionnels :

A. Besoins de disponibilité/fiabilité

Concernant le niveau de disponibilité qui doit être explicitement défini pour toute application critique.

Notre application sera opérationnel 24h/24h et 7j/7j vue qu'elle sera implantée sur un serveur d'application de la sorte que l'accès des utilisateurs est tout le temps assuré.

B. Besoins de sécurité :

Pouvant définir les niveaux d'accès possibles aux systèmes pour chacun des acteurs, garantir que l'accès aux espaces appropriés pour chaque acteur sera sécurisé par authentification par un login et un mot de passe, de plus les acteurs ne pourront manipuler que les informations qui sont à leurs charges.

C. Besoin d'efficacité :

Utilisation minimale de ressources matérielles (processeur, Mémoire, réseaux … etc.

Notre application tira profil de l'architecture client-serveur 3-tiers qui permet :

- Décharger le réseau, puisque seuls les requêtes et les résultats correspondants sont véhiculés.
- Economie en mémoire RAM et ROM sur les ordinateurs clients.
- Concentration de la puissance sur les serveurs (serveur d'application et base de données).

- Décharger le serveur des traitements propres aux applications (calcul, affichage, interaction avec l'utilisateur etc.). serveur dédié à la gestion de la base de données.

D. Besoins extensibilité :

Facilité avec laquelle de nouvelles fonctionnalités peuvent être ajoutées à un logiciel.

L'architecture client/serveur et l'utilisation d'un serveur d'application facilitera la mise à jour de l'ensemble de l'applicatif, de plus notre application sera un ensemble de fonctions attribuées à des groupes d'utilisateurs selon des permissions bien définis.

Cette architecture permettra d'un côté l'ajout de nouveau groupe d'utilisateur, et d'autre l'ajout de nouvelles fonctionnalités.

III.5.1.1.2- Acteurs et besoins fonctionnels :

Un acteur est une personne ou un système qui interagit avec le système étudié, en échangeant de l'information (en entrée et en sortie). On trouve les acteurs en observant les utilisateurs directs du système, les responsables de sa maintenance, ainsi que les autres systèmes qui interagissent avec lui. Ici, à partir de l'étude de l'existant que nous avons effectué au sein du service informatique au sein de la société Fond National d'Investissement, les acteurs de notre système sont:

Administrateur : il s'agit du chef du service informatique et des autres membres du service informatique. Ce sont les acteurs principaux du système car ils interagissent directement avec le système (consultation et mise à jour des données du système);

Utilisateur : il s'agit des personnes qui utilisent le matériel informatique. Ce sont des acteurs secondaires car ils n'ont pas d'interaction directe avec le système. Mais ils peuvent déclencher des évènements qui vont modifier l'état du système;

III.5.1.2- Diagramme de contexte :

Le diagramme de contexte est un modèle conceptuel de flux qui permet d'avoir une

vision globale des interactions entre le système et les liens avec l'environnement extérieur. Il permet aussi de bien délimiter le champ de l'étude. Pour notre cas le diagramme de contexte est donné par la figure suivant.

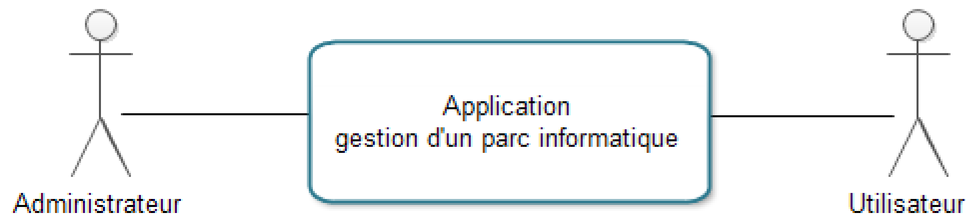


Figure III. 4 : Diagramme de contexte

III.5.1.3- DIAGRAMME DE CAS D'UTILISATION :

Un cas d'utilisation (User cas) modélise une interaction entre le système informatique à développer et un acteur interagissant avec le système, plus précisément, un cas d'utilisation décrit une séquence d'actions réalisées par le système qui produit un résultat observable pour l'acteur.

Il permet de décrire ce que le système devra faire ; sans spécifier le comment.

Plusieurs stratégies sont possibles pour l'identification des cas d'utilisation et des activités : procéder au regroupement par acteurs, par domaine fonctionnel. Pour notre étude nous allons procéder par regroupement par domaine fonctionnel, en faisant figurer les acteurs devant les paquetages d'activités auxquels ils ont accès.

Les différents packages que nous avons retenus sont illustrés dans les figures suivantes :

- **Diagramme global relatif au système:**

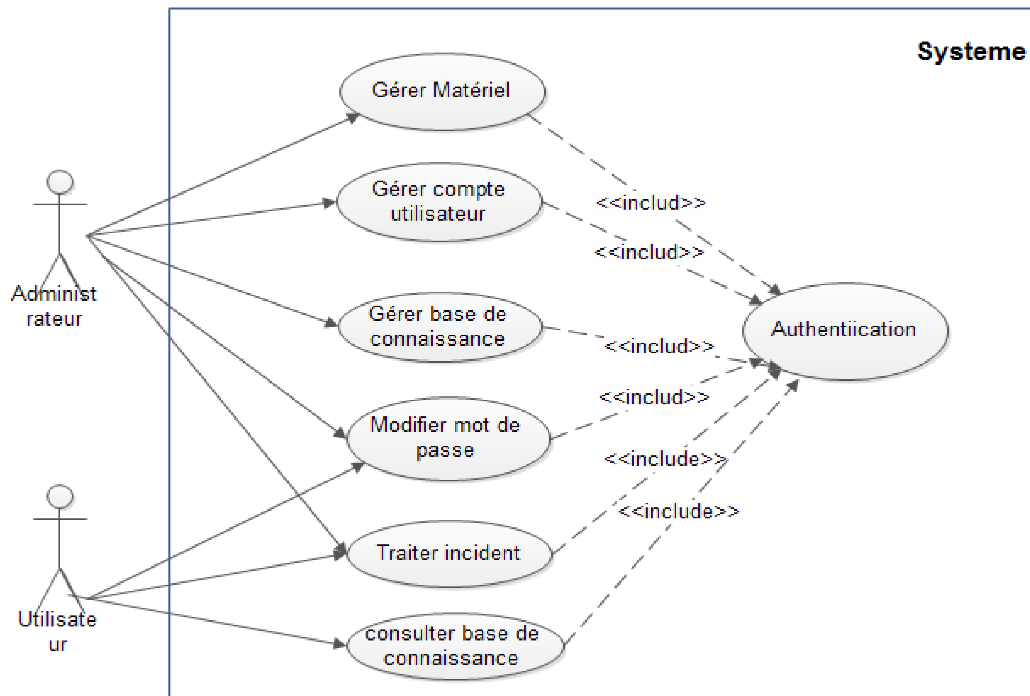


Figure III. 5 : Diagramme global relatif au système

- **Diagramme de cas d'utilisation « Gérer le matériel » :**

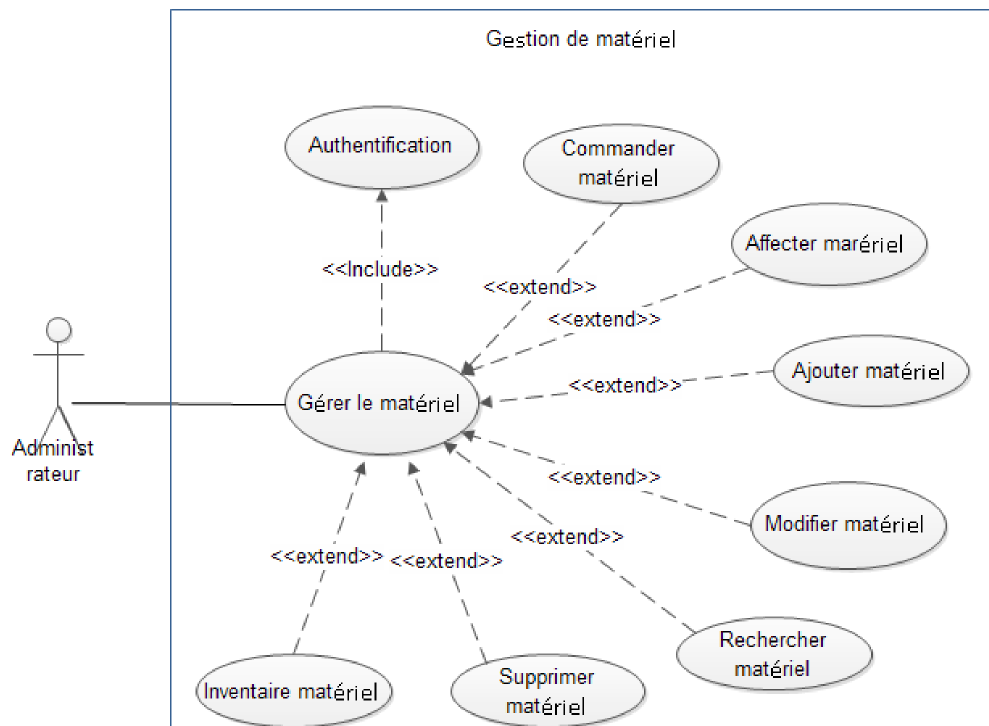


Figure III. 6 : Diagramme de cas d'utilisation « Gérer le matériel »

➤ **Diagramme de cas d'utilisation « Traitement ticket» :**

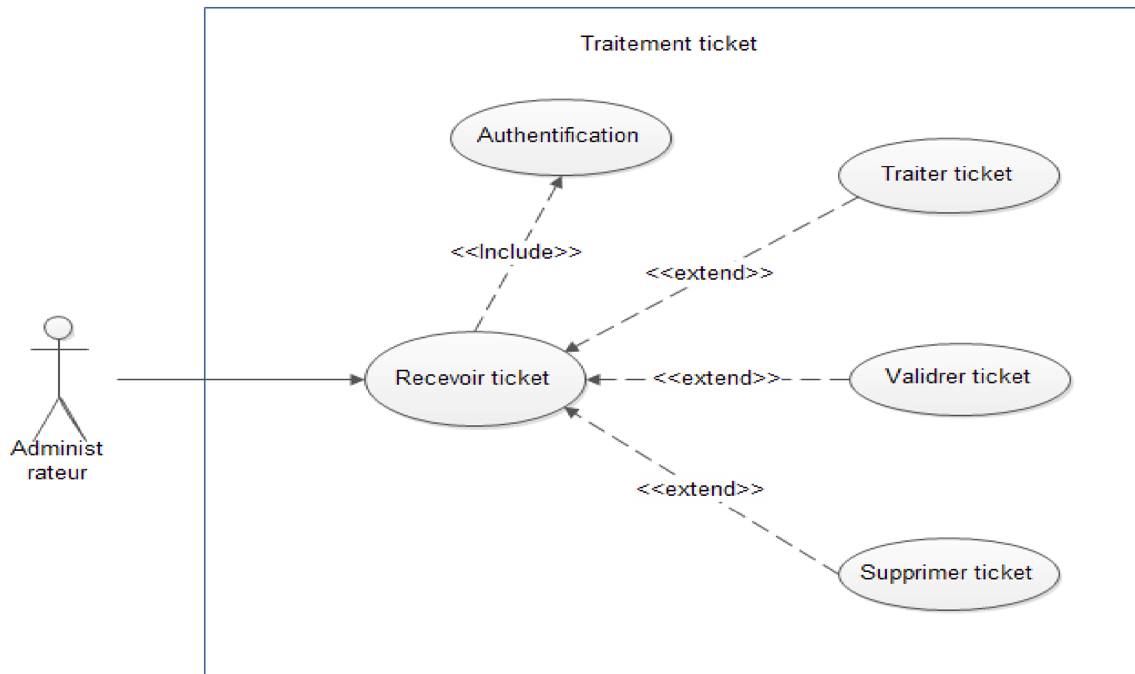


Figure III. 7 : Diagramme de cas d'utilisation « Traitement ticket»

➤ **Diagramme de cas d'utilisation « gestion compte utilisateur» :**

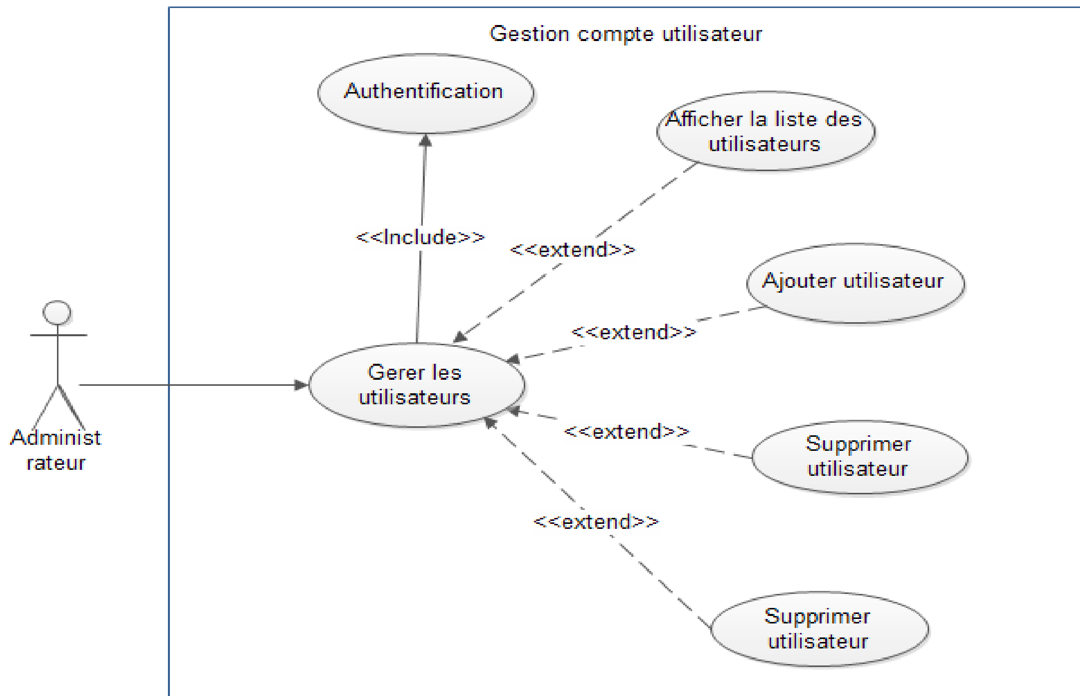


Figure III. 8 : Diagramme de cas d'utilisation « gestion compte utilisateur»

➤ **Diagramme de cas d'utilisation « Gérer BC » :**

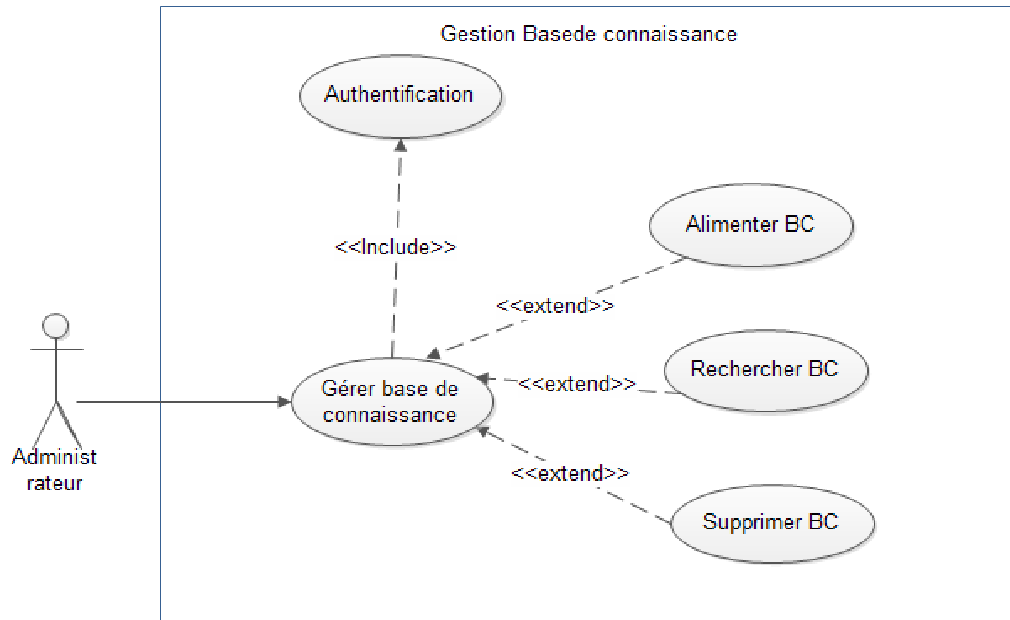


Figure III. 9 : Diagramme de cas d'utilisation « Gérer BC »

➢ **Diagramme de cas d'utilisation globale de l'utilisateur** :

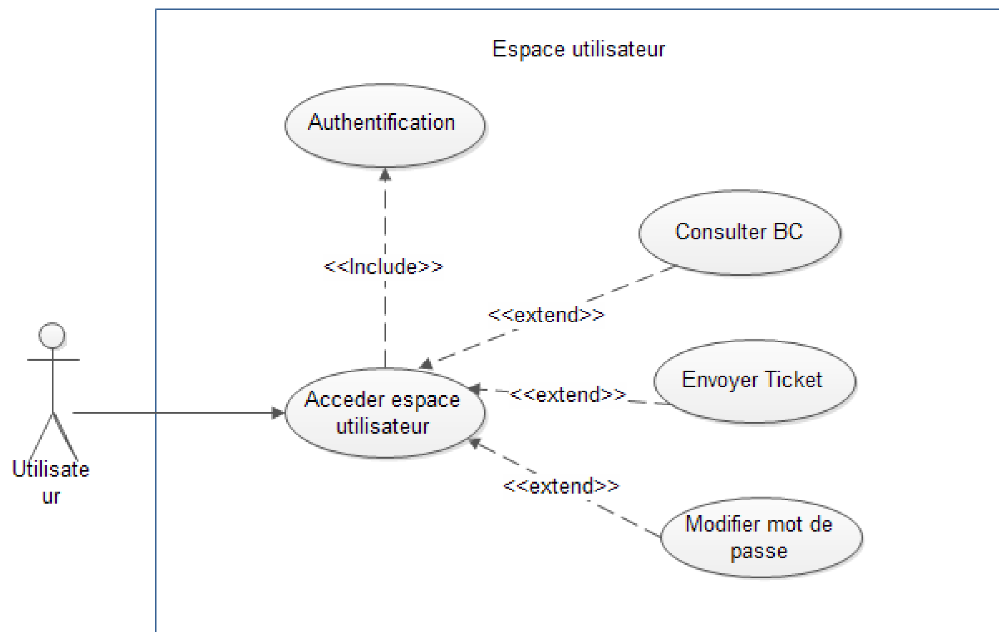


Figure III. 10 : Diagramme de cas d'utilisation globale de l'utilisateur

III.5.2- Conception :

La phase de conception permet de décrire de manière non ambiguë, le fonctionnement futur du système, afin d'en faciliter la réalisation.

Après avoir identifié les acteurs ainsi que les différentes fonctions du système à concevoir dans la partie analyse. Nous allons procéder comme suite :

- Pour chaque cas d'utilisation, élaboration du diagramme de séquence succédé d'un diagramme de scénario.
- Elaboration des diagrammes d'activités qui peuvent regrouper plusieurs cas d'utilisation et permettent de montrer l'organisation séquentielle des tâches.
- Elaboration des diagrammes de classes.

III.5.2.1- Diagrammes de séquences :

Définition:

Un diagramme de séquence montre une interaction présentée en séquence dans le temps. En particulier, il montre les objets (les instances) qui participent à l'interaction par leur ligne de

vie et les messages qu'ils échangent, présentés en séquence dans le temps. Il ne montre pas les associations entre les objets [15].

Diagramme de séquence pour le cas d'utilisation « identification » :

Cas d'utilisation : «Identification » :

Acteurs : Administrateur.

Scénario nominal :

1. L'utilisateur atteint la Formulaire d'identification.

2. Le système répond en affichant le formulaire d'identification.

3. L'utilisateur remplit le formulaire en tapant son login et mot de passe et valide en cliquant sur le bouton « Connexion »

4. Le système compare les données saisies par l'utilisateur avec celles de la base de données.

Enchaînement alternatif 1 :

A: Login et mot de passe corrects, le nom d'utilisateur est Administrateur.

Le scénario nominal démarre à l'étape «4» et on enchaine :

5. Le système charge l'espace approprié à l'administrateur.

Enchaînement alternatif 2 :

B : Login ou mot de passe correct, Login différent de administrateur.

Le scénario nominal démarre à l'étape «4» et on enchaine :

6. Le système vérifie l'état des sessions (Activée/Désactivée).

Enchaînement alternatif 2.1 :

C : la session est activée.

L'enchaînement alternatif 2 démarre à l'étape «6» et on enchaine :

7. Le système charge l'espace approprié à l'utilisateur correspondant.

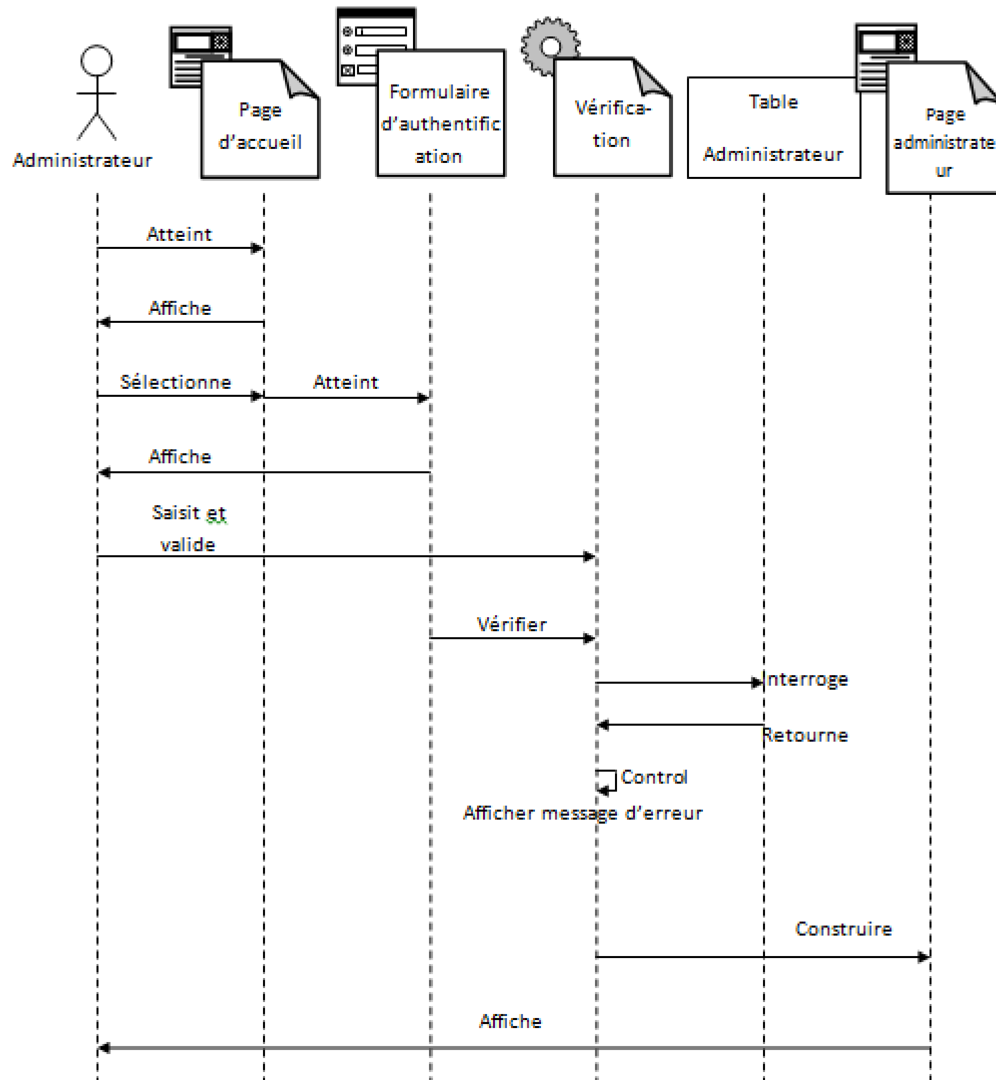


Figure III. 11 : Diagramme de séquence pour le cas d'utilisation « identification »

❖ **Diagramme de séquence pour le cas d'utilisation « consulter matériel » :**

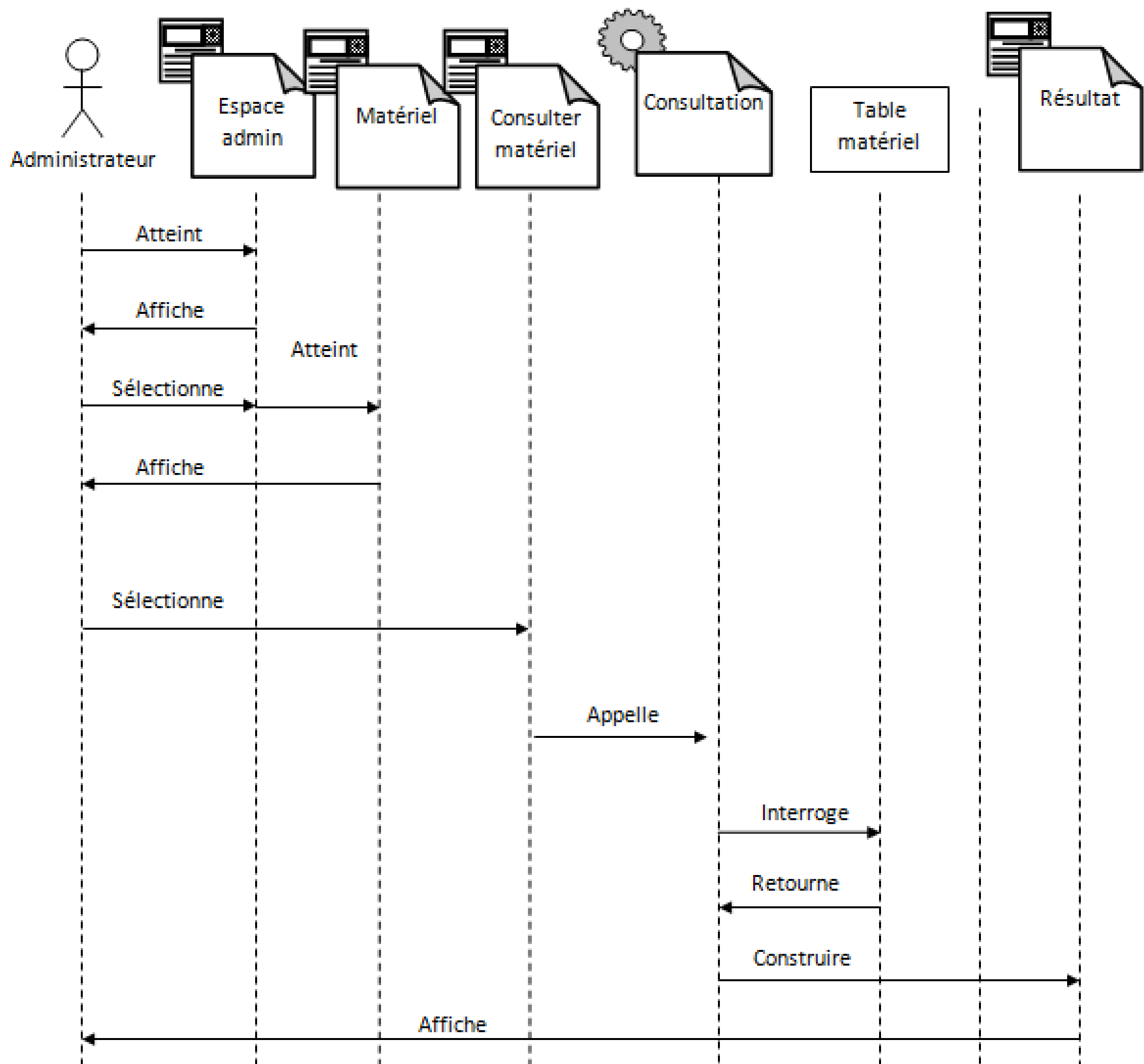


Figure III. 12 : Diagramme de séquence pour le cas d'utilisation « consulter matériel »

❖ **Diagramme de séquence pour le cas d'utilisation « ajouter matériel » :**

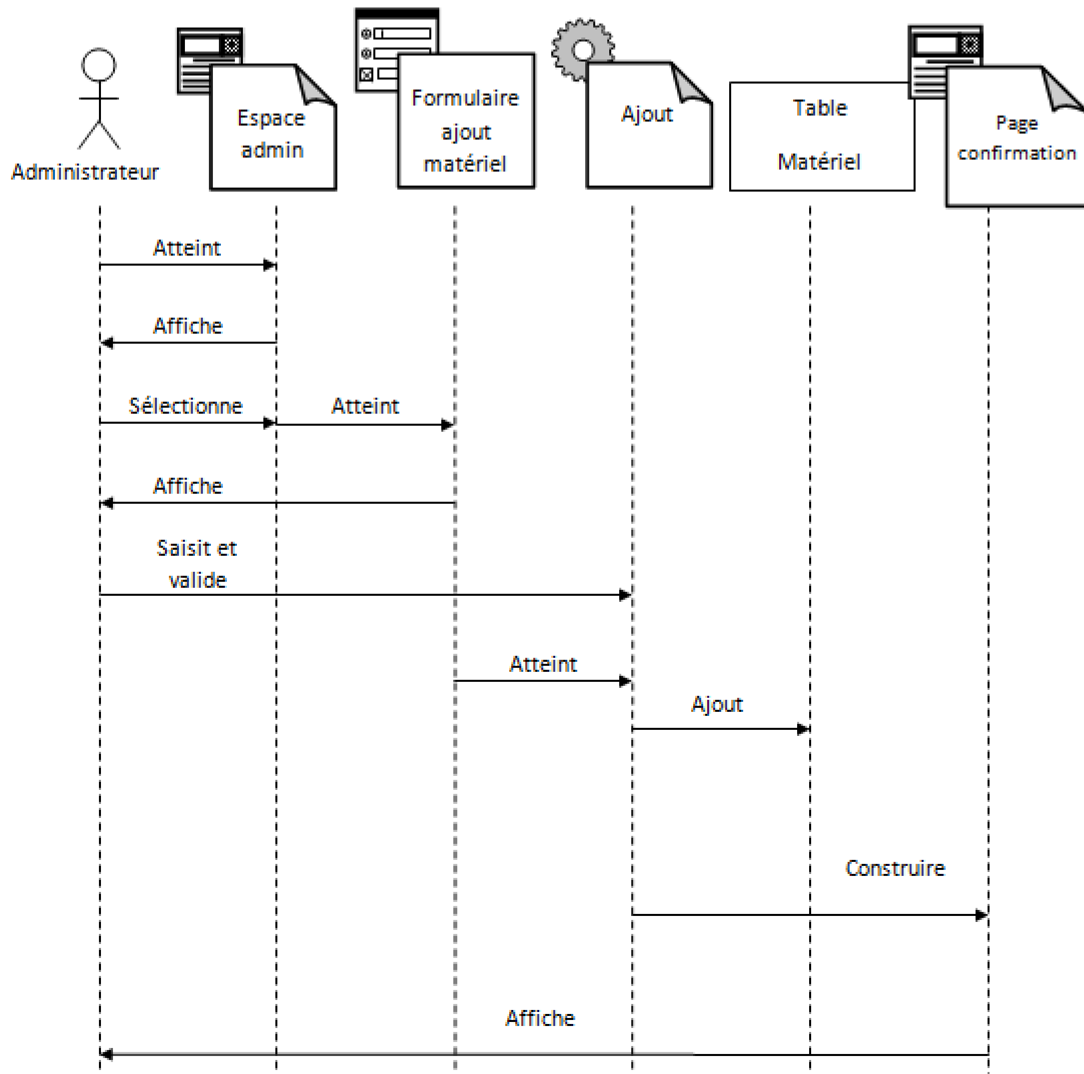


Figure III. 13 : Diagramme de séquence pour le cas d'utilisation «ajouter matériel »

Ajouter matériel

1. Le responsable matériel atteint son espace
2. Le responsable matériel sélectionne la rubrique « matériel »
3. Le système retourne le formulaire matériel
4. Le responsable matériel saisit et soumet les informations
5. Le système affiche une page de confirmation

❖ **Diagramme de séquence pour le cas d'utilisation « affecter matériel » :**

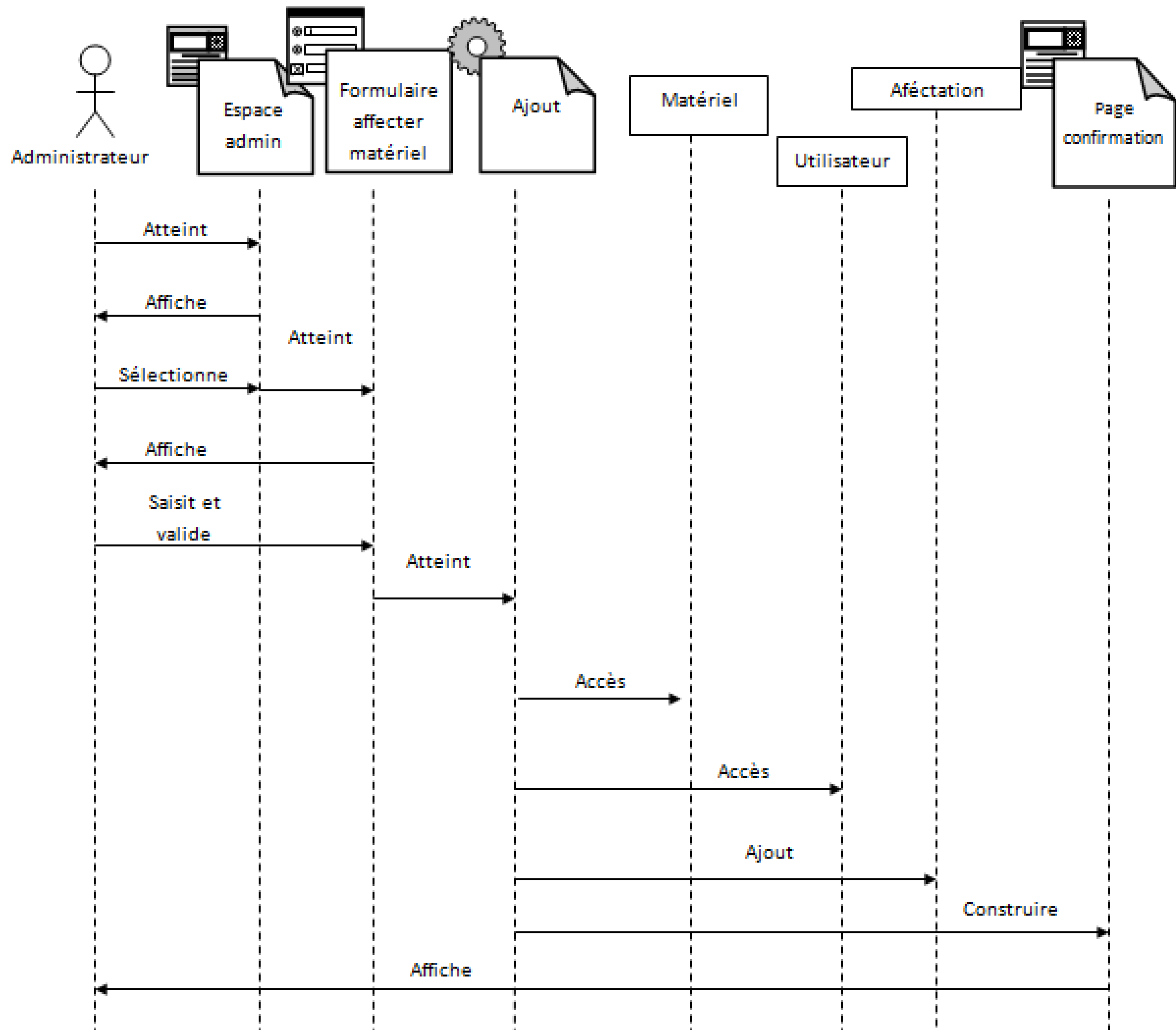


Figure III. 14 : Diagramme de séquence pour le cas d'utilisation «affecter matériel »

Affecter matériel à une base :

1. Le responsable matériel atteint son espace
2. Le responsable matériel sélectionne la rubrique « Affecter matériel »
3. Le système retourne le formulaire Affectation matériel
4. Le responsable matériel saisit et soumet les informations
5. Le système affiche une page de confirmation

❖ **Diagramme de séquence pour le cas d'utilisation « inventaire matériel »**

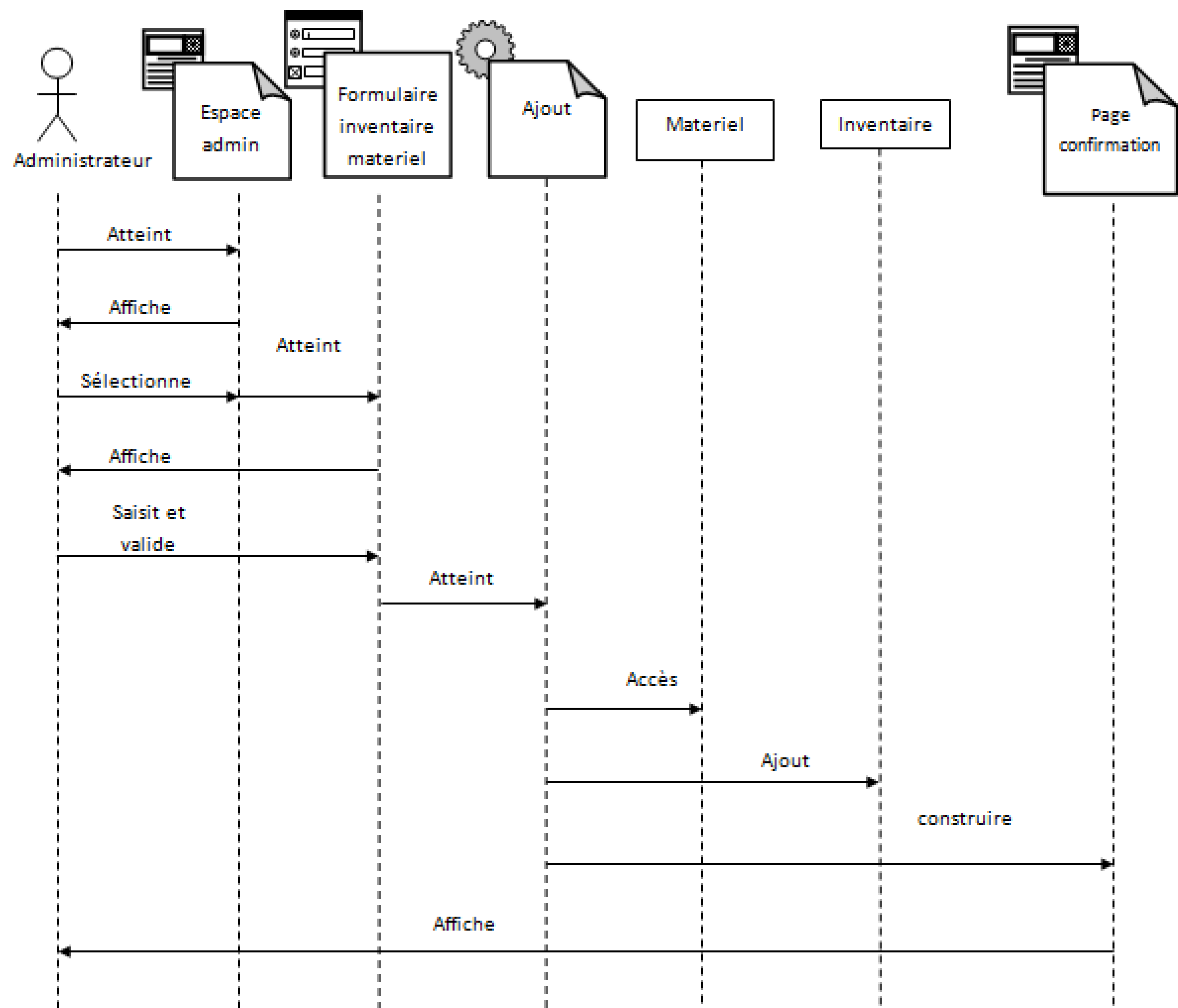


Figure III. 15 : Diagramme de séquence pour le cas d'utilisation « inventaire matériel »

Inventaire matériel

1. Le responsable matériel atteint son espace
2. Le responsable matériel sélectionne la rubrique « Inventaire matériel »
3. Le système retourne le formulaire Inventaire matériel
4. Le responsable matériel saisit et soumet les informations
5. Le système affiche une page de confirmation

III.5.2.2- Diagramme d'activités :

Un diagramme d'activité est une variante des diagrammes d'état/transition organisé par rapport aux actions et principalement destiné à représenter le comportement interne d'une méthode (la réalisation d'une méthode ou d'un cas d'utilisation).

Un diagramme d'activité représente l'état de l'exécution d'un mécanisme, sous la forme d'un déroulement d'étapes regroupées séquentiellement dans des branches parallèles de flots de contrôles.

Nous allons représenter les diagrammes d'activité pour quelques cas d'utilisation :

❖ **Identification :**

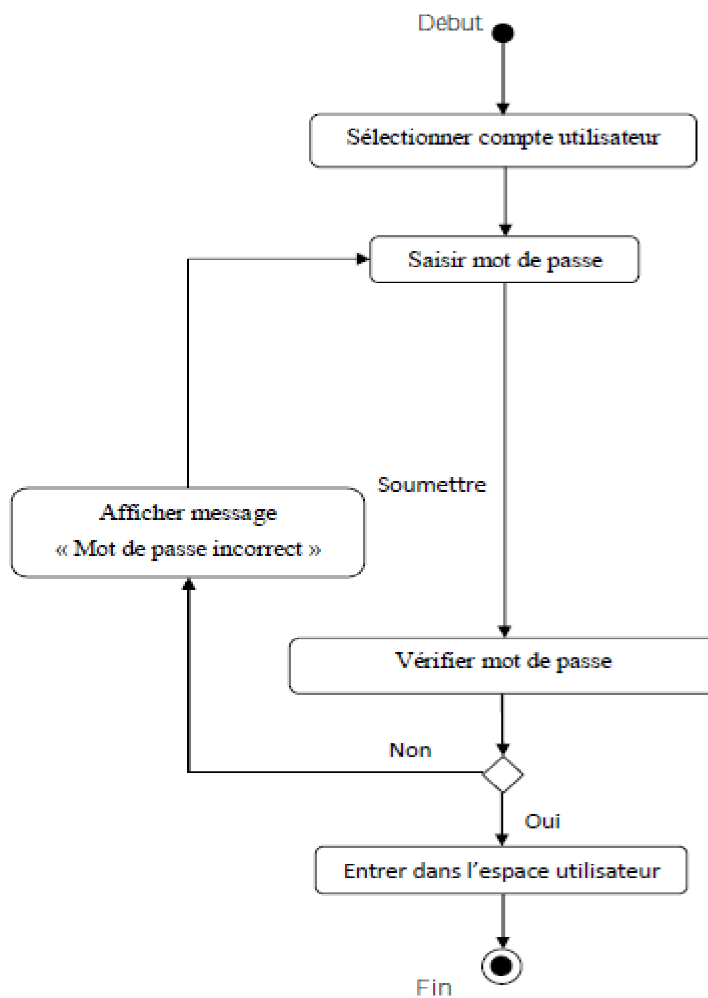


Figure III. 16 : Diagramme d'activité de l'opération « Identification ».

❖ **Ajouter matériel :**

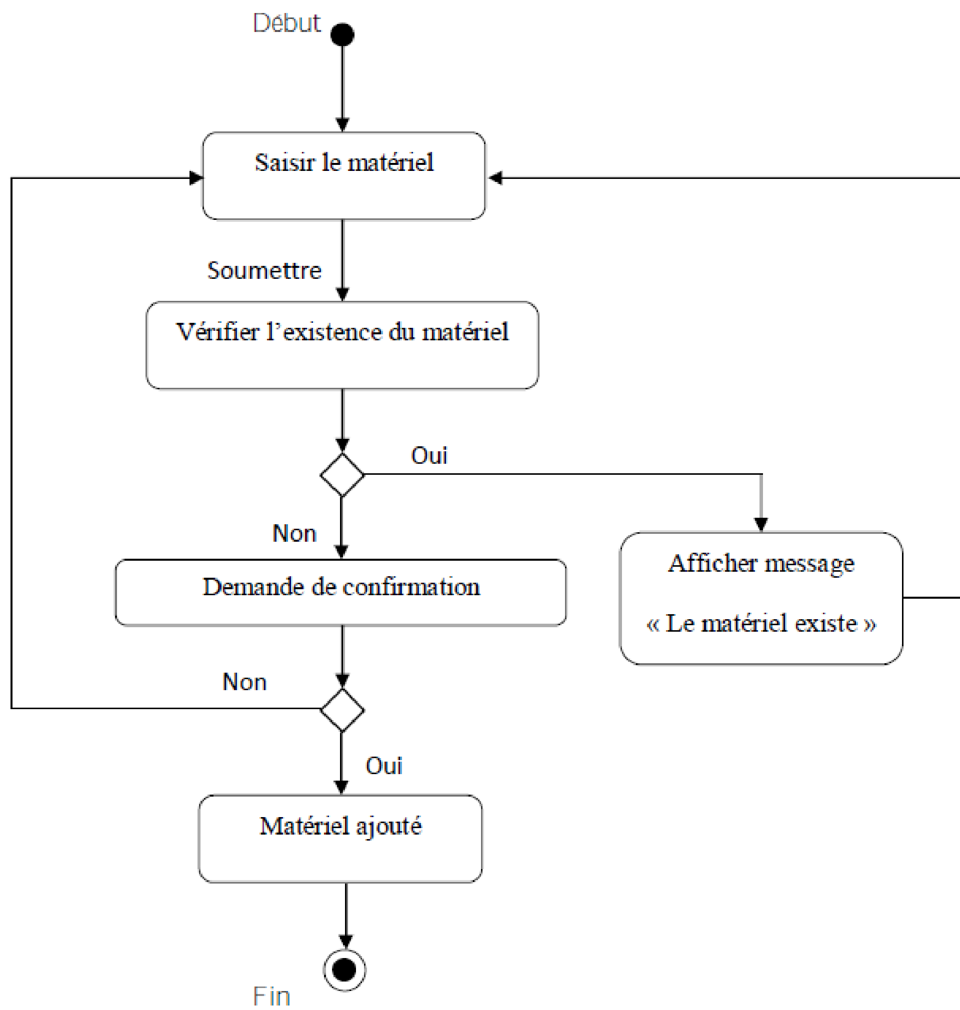


Figure III. 17 : Diagramme d'activité de l'opération « *Ajouter matériel* ».

❖ **Affecter matériel:**

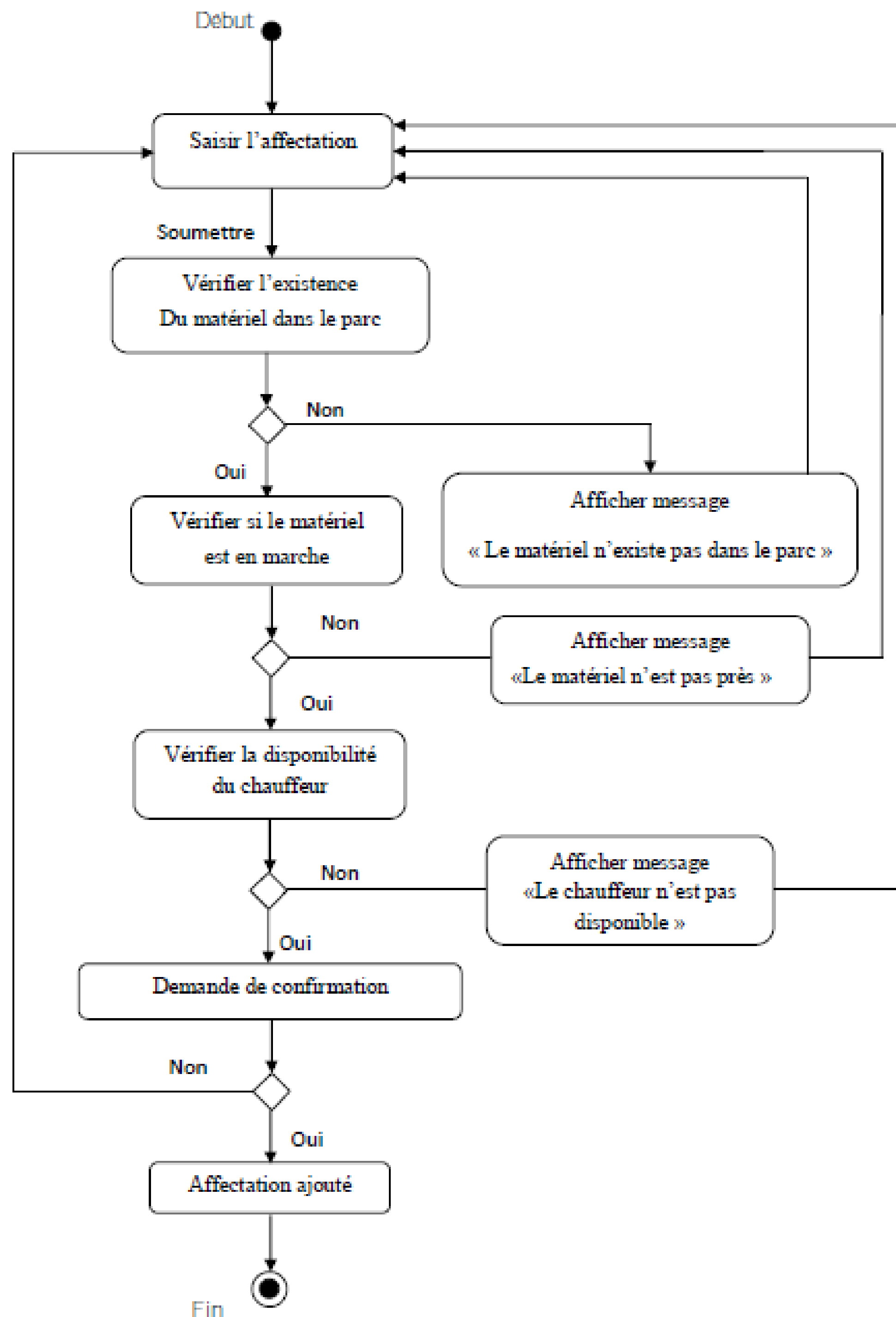


Figure III. 18 : Diagramme d'activité de l'opération « Affecter matériel ».

III.5.2.3- Digramme de classe :

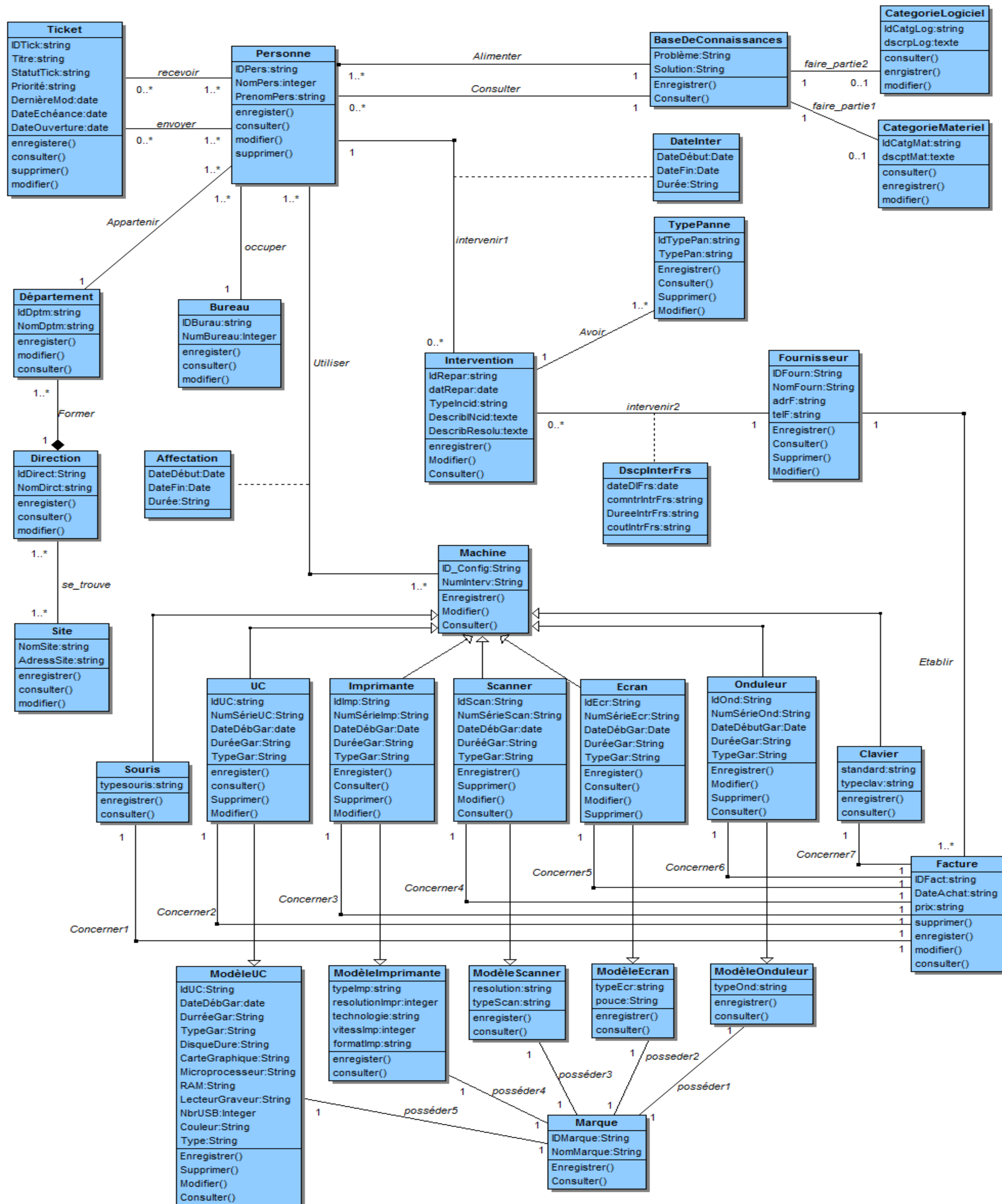


Figure III. 19 : diagramme de classe

Conclusion :

Dans ce chapitre, nous avons présenté une modélisation détaillée de notre future application de gestion du parc informatique. La prochaine étape consistera donc en la concrétisation du modèle de la solution que nous avons proposé, en d'autres termes, la réalisation des différents objets (Modèle de données, programmes, Forms, …etc.) qui la constituent.

Chapitre IV:

Implémentation et réalisation

Introduction :

Après avoir précisé dans le chapitre précédent les différentes étapes d'analyse et de conception, nous allons présenter dans ce dernier chapitre l'environnement de développement, les outils qui ont servi à la réalisation de notre application, et nous terminerons par la présentation de ses fonctionnalités à travers ses différentes interfaces.

IV.1-Environnement et outils de développement :

La création d'une application web avec Java EE s'effectue généralement à l'aide d'un Environnement de Développement Intégré, très souvent raccourci à l'anglaise IDE: C'est un logiciel destiné à faciliter grandement le développement dans son ensemble. S'il est possible pour certains projets Java de s'en passer, en ne se servant que d'un éditeur de texte pour écrire le code et d'une invite de commandes pour mettre en place l'application, ce n'est sérieusement plus envisageable pour la création d'une application web complexe.

❖ **Environnement**

Pour la réalisation de notre application nous avons fait appel à plusieurs outils :

- Système d'exploitation : Windows 7.
- Serveur Web : serveur Apache Tomcat (version 7.0.22).
- Le SGBD MySQL pour l'implémentation de notre base de données.
- Une interface PHPmyAdmin pour administrer la base de données.

IV.1.1-Le serveur Apache :

Pour faire fonctionner une application web Java EE, nous avons besoin de mettre en place un serveur d'applications. Il en existe beaucoup sur le marché: nous avons choisi d'utiliser Tomcat, car c'est un serveur léger, gratuit, libre, multiplateforme et assez complet pour ce que nous allons aborder. On le rencontre d'ailleurs très souvent dans des projets en entreprise, en phase de développement comme en production.

Tomcat est le serveur le plus répandu sur Internet, permettant la configuration de l'environnement d'exécution de pages web. Il s'agit d'une application fonctionnant à la base sur les systèmes d'exploitation de type Unix, mais il a désormais été porté sur de nombreux systèmes, dont Microsoft Windows grâce à sa conception modulaire (morceaux de code) qui correspond à différents aspects ou fonctions du serveur.

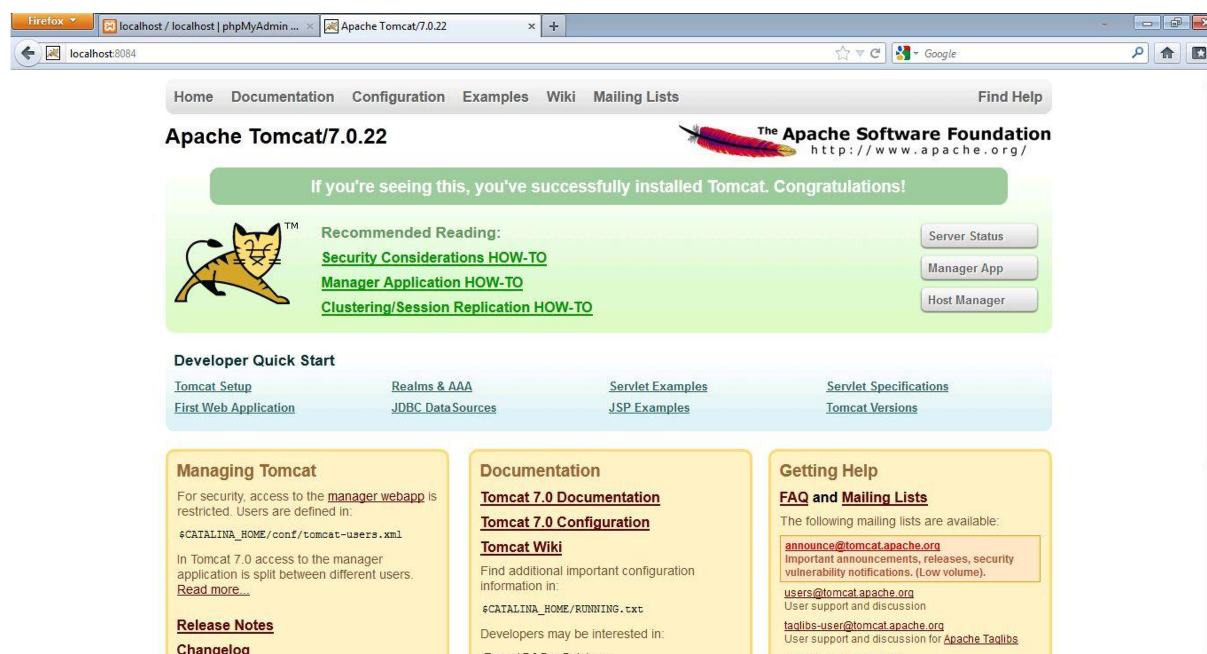


Figure IV.1 : Le serveur Apache Tomcat 7.0.22

IV.1.2-Le SGBD MySQL :

MySQL est un système de gestion de bases de données relationnelles (SGBDR) libre fonctionnant sous diverses plates-formes telles que UNIX, Linux et Windows, et permettant de manipuler des instructions adressées à la base de données Sous forme de requêtes SQL.

IV.1.3-L'interface PHPmyAdmin :

phpMyAdmin est une application web qui permet de gérer un serveur de bases de données MySQL dans un environnement multiutilisateur, cette interface permet également de donner à un utilisateur un accès à ses propres bases de données.

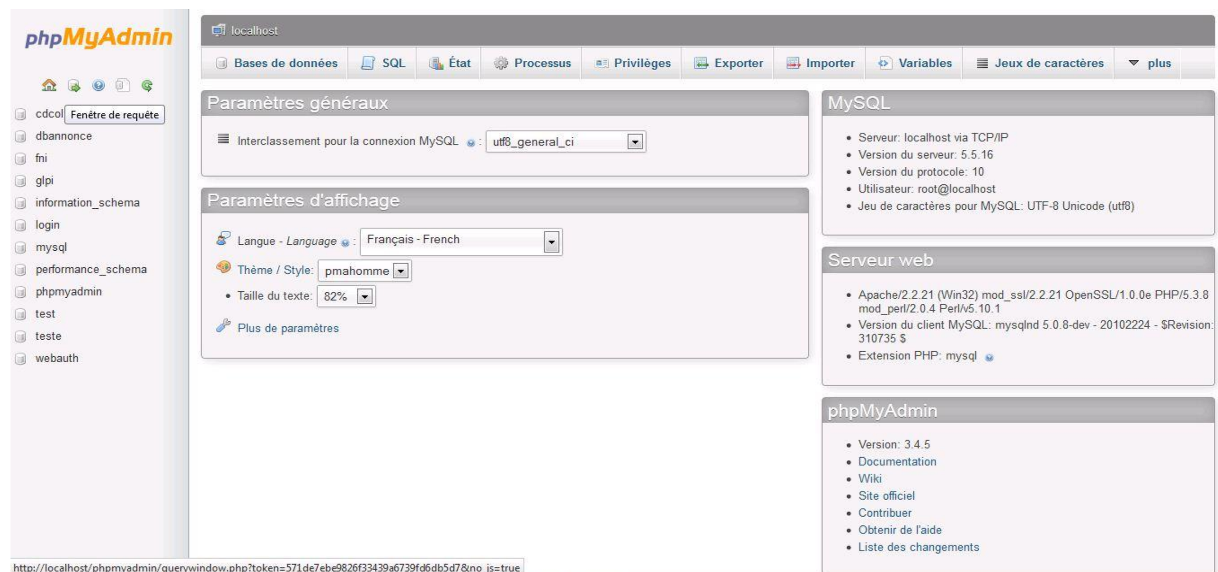


Figure IV.2: Interface PhpMyAdmin

- ❖ **Outils**

IV.1.4-NetBeans IDE 7.1.2 :

Est l'Environnement de Développement Intégré (IDE), nous avons préféré Netbeans à d'autres IDE vu sa simplicité et facilité d'utilisation ainsi la généralisation du code principale automatiquement des Jsps et Servlets et leur déploiement dans les fichiers XML sans avoir à le faire manuellement comme c'est le cas dans Eclipse.

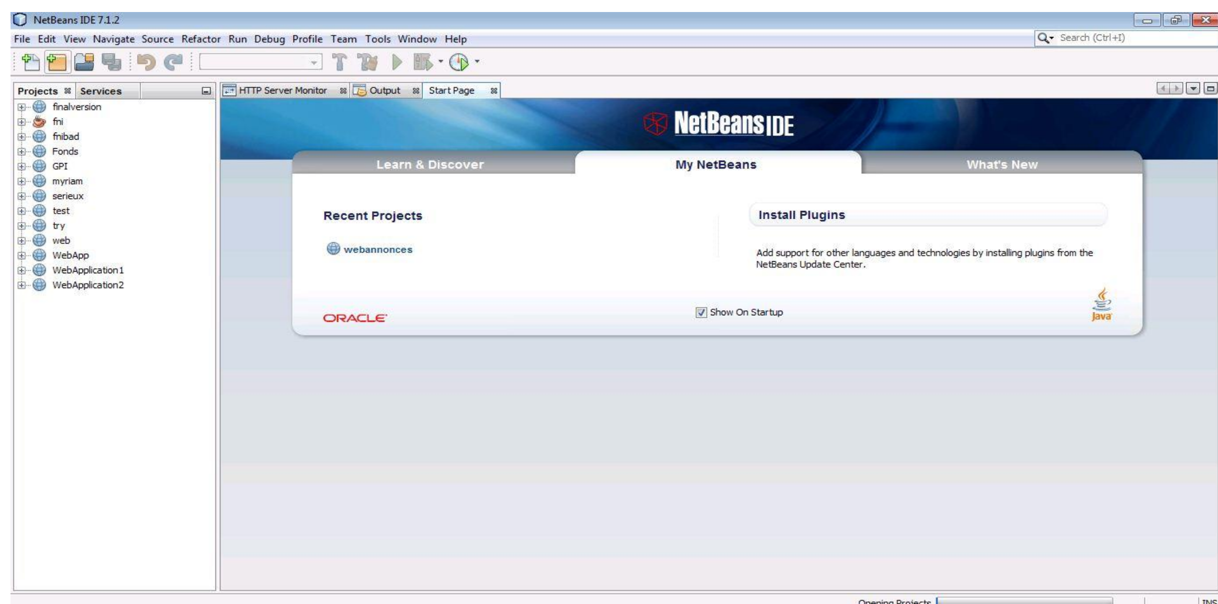


Figure IV.3: L'IDE NetBeans

IV.1.5-Macromedia Dreamweaver 8 :

Logiciel éditeur de texte créé par Macromedia (et géré maintenant par Adobe) permettant la conception de sites web. Dreamweaver fonctionne en mode WYSIWYG ou en mode code et dispose d'un Client FTP permettant le transfert des pages créées sur le serveur hébergeant les sites web.

Ce logiciel est très puissant d'utilisation .La partie Editeur de code du logiciel permet la modification rapide du contenu des pages, de leur mise en forme, de l'insertion d'images, de liens ou d'autres éléments habillant la page Web.

Nous avons choisi ce logiciel pour générer des pages JSP de qualité.

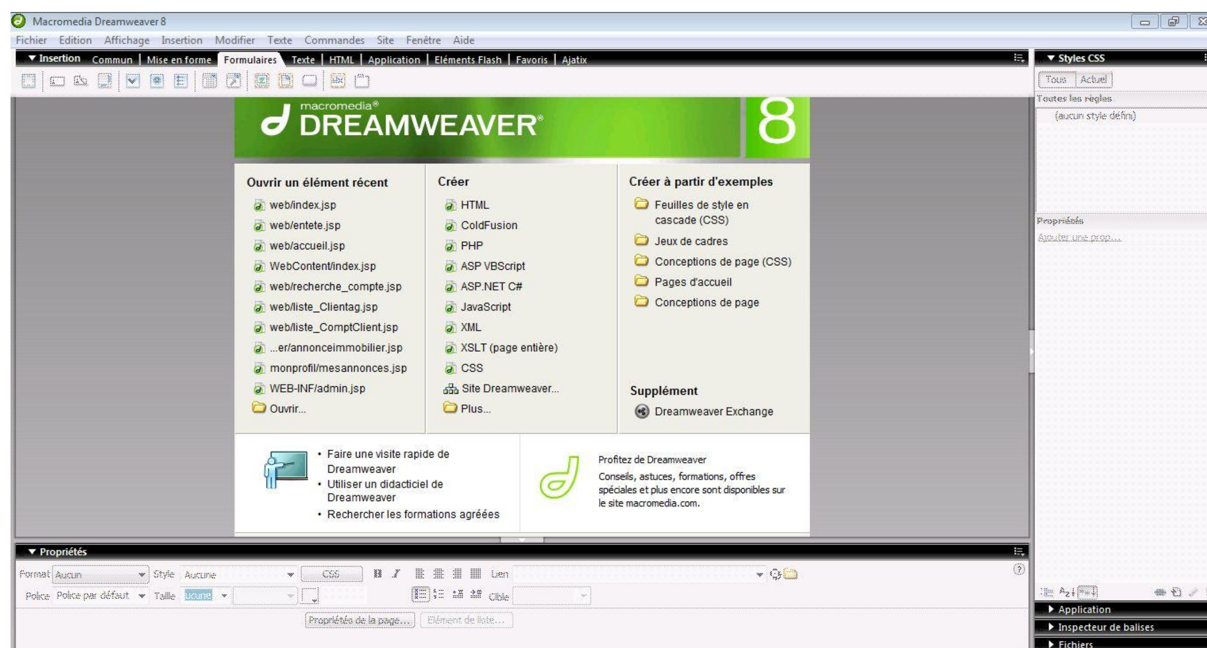


Figure IV.4: Dreamweaver

IV.1.6-Bouml :

BOUML est une série de logiciels comprenant un modeleur UML2 et plusieurs programmes externes dont des générateurs de code et reverse. BOUML ne couvre pas intégralement la norme UML2 à ce jour, mais le principal est déjà présent.

BOUML est développé en C++, ce qui en permet l'utilisation sous Windows et la constellation Unix (Linux, Solaris, Mac OS X etc.).

Le but de BOUML est de permettre une utilisation allant des besoins à la génération du code, les langages pris en compte à ce jour étant C++, Java, Php et IDL.

Nous avons utilisé BOUML pour la création des diagrammes de classes, séquences et cas d'utilisation.



FigureIV.5: Interface BOUML

IV.1.7-FireWorks :

Logiciel très utilisé chez les développeurs web, permet la création de bannières, arrières plan, boutons…etc.

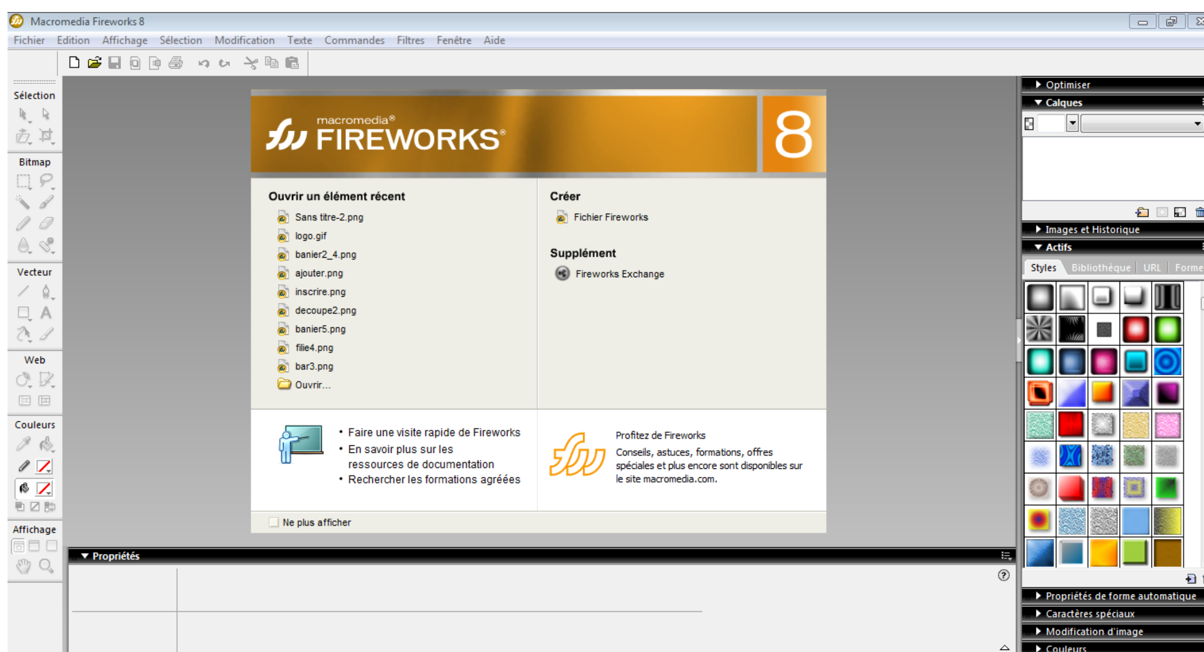


Figure IV.6: FireWorks

❖ Langages et architectures

IV.1.8-Le JAVA EE :

Le Java Enterprise Edition, comme son nom l'indique, a été créé pour le développement d'applications d'entreprises. Ses spécifications ont été pensées afin notamment de faciliter le travail en équipe sur un même projet: L'application est découpée en couches, et le serveur sur lequel tourne l'application est lui -même découpé en plusieurs niveaux. Pour faire simple, Java EE fournit un ensemble d'extensions au Java standard afin de faciliter la création d'applications centralisées.

Il faut bien se rendre compte qu'à l'origine, Java EE permet plus ou moins de coder son application comme on le souhaite : en d'autres termes, on peut coder n'importe comment ! Or on sait que dans le développement en entreprises implique entre autres que l'on puisse être amené à travailler à plusieurs contributeurs sur un même projet ou une même application (travail en équipe); que l'on puisse être amené à maintenir et corriger une application que l'on n'a pas créée soi-même; que l'on puisse être amené à faire évoluer une application que l'on n'a pas créée soi-même.

Pour toutes ces raisons, il est nécessaire d'adopter une architecture plus ou moins standard, que tout développeur peut reconnaître, c'est-à-dire dans laquelle tout développeur sait se repérer.

Il a très vite été remarqué qu'un modèle permettait de répondre à ces besoins, et qu'il s'appliquait particulièrement bien à la conception d'applications Java EE: le modèle MVC (Modèle-Vue-Contrôleur).

Il découpe littéralement l'application en couches distinctes, et de ce fait impacte très fortement l'organisation du code. Voici dans les grandes lignes ce qu'impose MVC :

Tout ce qui concerne le traitement, le stockage et la mise à jour des données de l'application doit être contenu dans la couche nommée " Modèle" (l e M de MVC) ;

Tout ce qui concerne l'interaction avec l'utilisateur et la présentation des données (mise en forme, affichage) doit être contenu dans la couche nommée " Vue" (l e V de MVC) ;

Tout ce qui concerne le contrôle des actions de l'utilisateur et des données doit être contenu dans la couche nommée " Contrôle" (l e C de MVC).

Modèle : des traitements et des données

Dans le modèle, on trouve à la fois les données et les traitements à appliquer à ces données. Ce bloc contient donc des objets Java d'une part, qui peuvent contenir des attributs (données) et méthodes (traitements) qui leurs sont propres, et un système capable de stocker des données d' autre part. La complexité du code ici dépendra bien évidemment de la complexité des traitements à effectuer l'application.

Vue : des pages JSP

Une page JSP est destinée à la vue, exécutée côté serveur et permet l'écriture de gabarits (pages en langage "client" comme HTML, CSS, Java script, XML, etc.). Elle permet au concepteur de la page d'appeler de manière transparente des portions du code Java, via des balises et expressions ressemblant fortement aux balises de présentation HTML .

Contrôleur : des servlets

Une servlet est un objet qui permet d'intercepter les requêtes faites par un client, et qui peut personnaliser une réponse en conséquence. elle fournit pour cela des méthodes permettant de scruter les requêtes HTTP. Cet objet n'agit jamais directement sur les données, il faut le voir comme un simple aiguilleur: il intercepte une requête issue d'un client, appelle éventuellement des traitements effectués par le modèle, et ordonne en retour à la vue d'afficher le résultat au client.

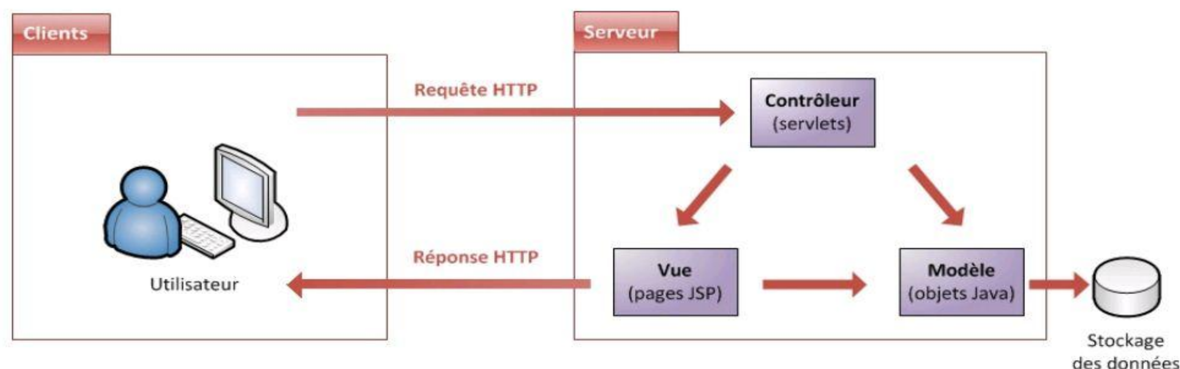


Figure IV.7: L'architecture MVC

Beans :

Un bean est un simple objet Java qui suit certaines contraintes, et représente généralement des données du monde réel.

Voici un récapitulatif des principaux concepts mis en jeu. Je vous donne ici des définitions plutôt abstraites, mais il faut bien passer par là :

Les propriétés : un bean est conçu pour être paramétrable. On appelle "propriétés" les champs non publics présents dans un bean. Qu'elles soient de type primitif ou objets, les propriétés permettent de paramétrer le bean, en y stockant des données.

La sérialisation: un bean est conçu pour pouvoir être persistant. La sérialisation est un processus qui permet de sauvegarder l'état d'un bean, et donne ainsi la possibilité de le restaurer par la suite. Ce mécanisme permet une persistance des données, voire de l'application elle-même.

La réutilisation: un bean est un composant conçu pour être réutilisable. Ne contenant que des données ou du code métier, un tel composant n'a en effet pas de lien direct avec la couche de présentation, et peut également être distant de la couche d'accès aux données. C'est cette indépendance qui lui donne ce caractère réutilisable.

L'introspection: un bean est conçu pour être paramétrable de manière dynamique. L'introspection est un processus qui permet de connaître le contenu d'un composant (attributs, méthodes et événements) de manière dynamique, sans disposer de son code source. C'est ce processus couplé à certaines règles de normalisation qui rend possible une découverte et un paramétrage dynamique du bean.

IV.1.9- Connexion à la base de données MySQL via JDBC :

JDBC est l'acronyme de Java Data Base Connectivity. C'est l'API standard pour permettre un accès à une base de données. Son but est de permettre de coder des accès à une base de données en laissant le code le plus indépendant de la base de données utilisée.

C'est une spécification qui définit des interfaces pour se connecter et interagir avec la base de données (exécution de requêtes ou de procédures stockées, parcourt des résultats des requêtes de sélection, ...).

L'implémentation de ces spécifications est fournie par des tiers, et en particulier les fournisseurs de base de données, sous la forme de Driver.

IV.1.10- Le langage de requête SQL :

Pour communiquer avec une base de données, on a besoin de lui envoyer des commandes ou instructions appelées requêtes. Que ce soit pour la création, la suppression d'une table, la modification, l'insertion ou la sélection de données, le langage standard de requêtes est SQL.

SQL (Standard Query Language) est un langage permettant d'interroger les bases de données de manière simple. Il est doté d'une syntaxe particulière que l'on doit respecter pour que la communication avec la base se passe au mieux. Son succès est dû essentiellement à sa simplicité et au fait qu'il énonce des requêtes en laissant le SGBD responsable de la stratégie d'exécution.

SQL couvre les trois fonctions indispensables à la mise en œuvre et à l'exploitation de bases de données relationnelles :

- la création des données ;
- la manipulation des données ;

IV.2-Fonctionnalités :

IV.2.1-Authentification :

L'application ne présente aucun accès libre aux utilisateurs, mais nécessite une authentification pour pouvoir l'exploiter. La figure ci-dessous est l'interface visible en

premier lieu et nous dirige vers notre page appropriée selon la catégorie qui peut s'agir d'un utilisateur ou d'un administrateur et ce, à l'aide d'une variable booléenne qui fait un test sur le champ (id_cat) de la table catégorie, si l'identifiant égale à 1, l'administrateur pourra naviguer dans son espace librement sinon, l'accès sera offert aux utilisateurs.

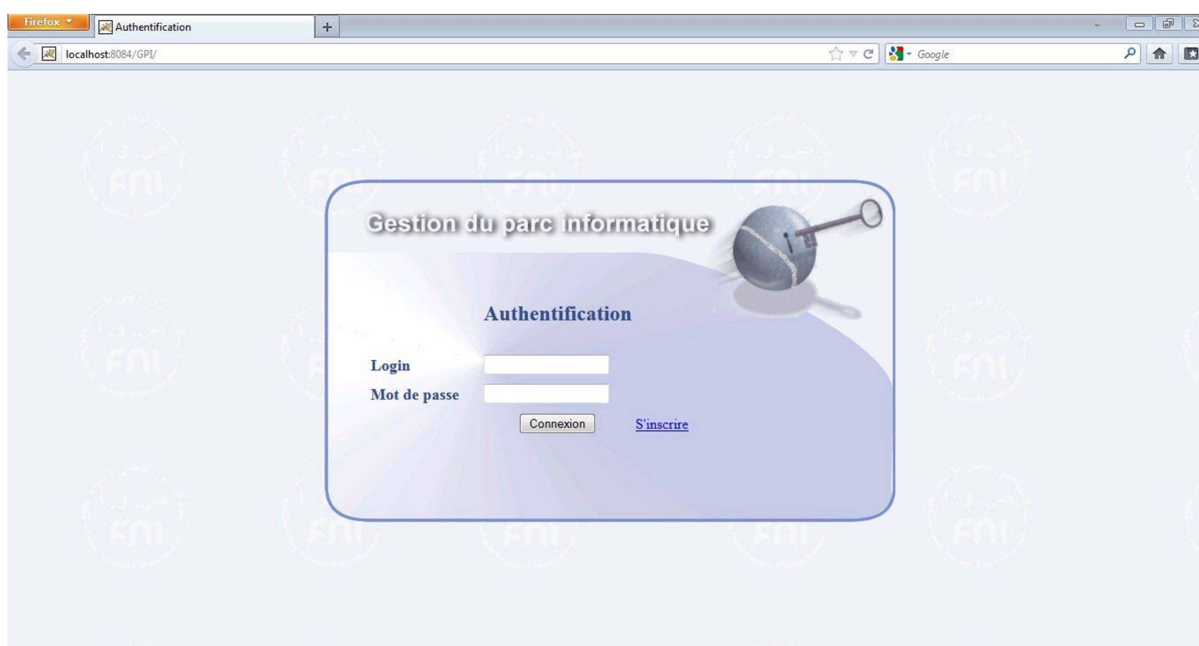


Figure IV.8 : Vue Authentification

IV.2.2-Espace Administrateur :

Ci-dessous la vue de l'administrateur après authentification.



Figure IV.9: Vue Espace Administrateur

IV.2.3-Inscription :

A l'instar de plusieurs sites publiés sur le web, notre application offre la possibilité à un visiteur sur le réseau de devenir membre, et pouvoir ainsi bénéficier de ce qu'offre l'application.

Ci-dessous le formulaire d'inscription.

Figure IV.10 : Vue Inscription

IV.2.4-Ajout :

Des formulaires adaptés à chaque type de matériels sont mis en œuvres. Un contrôle de saisie des champs obligatoires et numériques permet de ne transmettre à la base que des données valides, sinon le formulaire est réaffiché avec les champs pré remplis et un message d'erreur indiquant la nature de l'erreur.

Figure IV.11 : Vue Ajout modèle UC

IV.2.5-Affichage :

A l'aide des interfaces d'affichage, nous pouvons visualiser toutes les caractéristiques d'un matériel donné et toutes les informations d'un utilisateur.

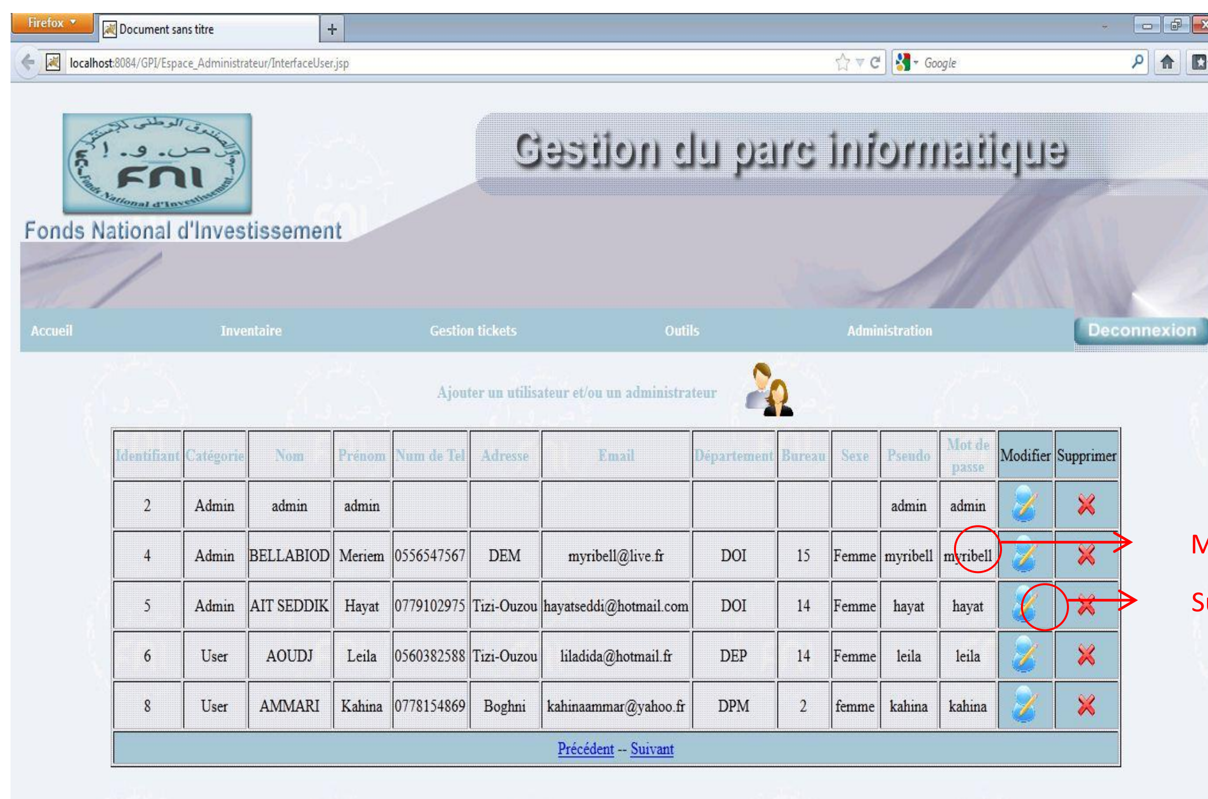


Figure IV.12 : Vue Liste User

IV.2.6-Modification :

Grâce à cette option, l'administrateur peut à tout moment apporter des modifications sur les caractéristiques d'un matériel. Cela se fait sans avoir besoin de saisir tous les champs, grâce à un formulaire pré rempli avec les anciennes caractéristiques, il suffit donc de modifier les champs qui nous intéressent.

IV.2.7-Suppression :

Supprimer les entrées qui ne sont plus valides ou expirées.

Conclusion :

Dans ce chapitre nous avons présenté les outils utilisés pour le développement de notre application et on a donné quelques explications sur les fonctionnalités de l'application par une illustration de certaines interfaces.

Nous avons essayé de concevoir une application qui se rapproche le plus possible des outils de gestion du parcs informatiques qu'on trouve dans le commerce et digne de ce nom, tout en répondant au mieux au cahier des charges défini par le maitre d'ouvrage.

Conclusion

Conclusion générale :

Notre projet porte sur la mise en place d'une application client/serveur pour l'automatisation des activités de gestion et de suivi du matériel informatique au niveau du FNI.

L'application assurera un environnement interactif afin de rendre les tâches de gestion accessibles aux différents acteurs dans le but de maîtriser les différentes fonctions de base du système.

Le stage que nous avons effectué au sein du service DOI du FNI, nous a appris beaucoup de choses.

Humainement, grâce à la communication avec le personnel du service DOI pour établir les besoins et voir de près les tâches complexes auxquelles ils font face et afin de répartir les tâches et gérer nos différences. Mais aussi techniquement, ce projet nous a permis d'appliquer nos acquis dans un contexte professionnel et nous a amené à découvrir certains points non développés en cours.

En plus de l'expérience, ce projet à travers les étapes de mise en œuvre notamment l'analyse, la conception et la réalisation, nous a conduit à l'utilisation des outils de conception « UML » et la réalisation de l'application avec le langage java avec tous ses outils et son environnement et de se familiariser avec eux.

La conception orientée objet du système proposé garanti de ce fait une souplesse dans l'intégration et l'extension du système.

Quoique notre application offre plusieurs services et rend aisée les tâches de gestion du parc, à titre d'exemples, gestion du mouvement de matériels, gestion des interventions aux incidents, possibilité d'inventorier ainsi que l'implémentation d'un module de sécurité qui permet d'attribuer les droits d'accès aux utilisateurs ...etc.

En fin, nous espérons que ce modeste travail sera de grand intérêt pour l'organisme du FNI et un guide pour les prochaines promotions.

Références bibliographies

Référence Bibliographies:

Les ouvrages et Articles :

Traces	Ouvrages et articles
[1]	Richard Grin, Introduction à l'utilisation des réseaux informatiques, Université de Nice Sofia-antripolis.
[2]	«les réseaux », auteur pujolle, Edition 2003.
[4]	Guy Rujolle, « les réseaux »
[5]	Pascal Nicolas « cours de réseaux maitrise d'informatique », université d'Angers
[6]	Jean François Pillon, « tout sur les systèmes d'informations », édition Dunod, 2006
[7]	Fronçois Marie Colonna, « l'architecture client-serveur octobre 2002.
[8]	De Merise à UML, préface Hubert TARDIEU Edition EYROLLES
[18]	Pascal Roques, « UML2 par la pratique étude cas et exercice corrigés », 5eme édition eyrolles
[19]	Jim Conallen, " Concevoir des applications Web avec UML ", Edition : Eyrolles, 2000.
[20]	Chantal Morley,jeam Hugues,Bernard leblanc: "UMl pour

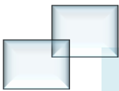
	l'analyse d'un système d'information ", Edition : Dunod, 2000.
[21]	Michel Lai,"UML la notation unifier de modèlisation objet", Edition : Dunod, 2000.
[22]	Grady Booch et al, " Le Guide de l'utilisateur UML ", Edition : Eyrolles, 2003.

Les supports de cours, mémoires, lien Internet :

[9]	**Cours réseaux : Le modèle OSI Frédéric Jacquenod**
[10]	Cours réseaux <<Université Mouloud MAMMERI, département informatique.Tizi-ouzou>>
[11]	TAZBOUDJT Said, TOUMERT Mouhamed. Conception et réalisation d'une application Client/Serveur 3-tiers sous Oracle, cas : Gestion des Ressources Humaines de l'ENIEM. Mémoire d'ingénieur UMMTO, 2007.
[12]	http://www-gtr.iutv.univ-paris13.fr/ <<R.Kanawati, cours réseaux>>
[13]	http://www.grappa.univ-lille3.fr <<

	réseaux architecture et protocoles>>
[14]	les réseaux informatiques D.Lalot : http://ciscam.univ-aix.fr/dotech/reseaux.pdf
[15]	www.commentca marche.net
[16]	FNI/BAD Dossier kdb, ORGANISATIONBAD 3.doc, 29 octobre 2009
[17]	Site de FNI : www.fnibad.gov

Annexes



Annexe A.UML :

A.1. Introduction :

Pour faire face à la complexité croissante des systèmes d'information, de nouvelles méthodes et outils ont été créés. La principale avancée des quinze dernières années réside dans la

Programmation Orientée Objet (P.O.O.).

Devant le foisonnement de nouvelles méthodes de conception« orientée objet », l'Object

Management Group (OMG) a eu comme objectif de définir une notation standard utilisable dans les développements informatiques basés sur l'objet. C'est ainsi qu'est apparu UML

(Unified Modeling Language « langage de modélisation objet unifié »), qui est issu de la fusion des méthodes Booch, OMT (Object Modelling Technique) et OOSE (Object Oriented

Software Engineering).

Issu du terrain et fruit d'un travail d'experts reconnus, UML est le résultat d'un large consensus. De très nombreux acteurs industriels de renom ont adopté UML et participent à son développement.

En espace d'une poignée d'années seulement, UML est devenu un standard incontournable, dont le but est de formaliser la communication, afin de dépasser l'ambiguïté de langue naturelle. UML permet donc de standardiser et de normaliser le processus de modélisation lors de la phase de conception. En effet, UML applique la croyance populaire selon laquelle, en guise d'explication, un petit schéma vaut souvent mieux qu'un long discours. La documentation liée à un projet devient alors universelle.

Cette annexe va nous permettre d'apprendre ce qui est UML et de maîtriser les aspects syntaxiques de langage.

A.2. Définition d'UML :

UML est un langage standard adapté à la modélisation des systèmes, depuis les systèmes informatiques d'entreprise jusqu'aux applications distribuées basées sur le web, en passant

par les systèmes temps réel embarqués. C'est un langage très expressif qui couvre toutes les perspectives nécessaires au développement puis au déploiement de tels systèmes.

UML est un langage qui permet de représenter des modèles, mais il ne définit pas le processus d'élaboration d'un modèle. Cependant, dans le cadre de la modélisation d'une application informatique, les auteurs d'UML préconisent d'utiliser une démarche itérative et

incrémentale, guidée par les besoins des utilisateurs du système, Centrée sur l'architecture logicielle. [19]

A.3. Modélisation avec UML :

A.3.1 Définition d'un modèle :

Un modèle est une simplification et/ou une abstraction de la réalité. Il doit aider à mieux comprendre, percevoir les relations et les interactions à l'intérieur du système. Il doit permettre de visualiser les conséquences de modifications apportées au système, de visualiser également les raisons de comportement du système par rapport à une situation donnée. C'est donc un guide pour construire un système stable et fiable. Le modèle doit également aider à documenter le système construit. [20]

A.3.2 Modélisation UML :

UML fournit une panoplie d'outils permettant de représenter l'ensemble des éléments du monde objet (classes, objets, etc.) ainsi que les liens qui les relient. Toutefois, étant donné qu'une seule représentation est trop subjective, UML fournit un moyen astucieux permettant de représenter diverses projections d'une même représentation grâce aux vues. Une vue est constituée d'un ou plusieurs diagrammes.

On distingue deux types de vues :

◊ **Les vues statiques :** représentant le système physiquement

- Diagrammes d'objets,
- Diagrammes de classes,
- Diagrammes de cas d'utilisation,

- Diagrammes de composants,
- Diagrammes de déploiement.

◊ **Les vues dynamiques :** montrant le fonctionnement du système

- Diagrammes de séquence,
- Diagrammes de collaboration,
- Diagrammes d'états de transition,
- Diagrammes d'activités.

A.4. Concepts d'UML :

UML se veut être une notation simple, précise, et homogène, permettant un bon rendu visuel.

Elle décrit la réalité plutôt que le processus de réalisation. [21]

A.4.1. Acteur :

Un acteur représente un rôle joué par une entité externe (utilisateur humain, dispositif matériel ou autre système) qui interagit directement avec le système étudié.

On dénombre 4 types d'acteurs :

Acteurs principaux: Ceux qui interagissent normalement avec le système (client de la billetterie automatique par exemple),

Acteurs secondaires: Ceux qui réalisent des tâches administratives ou de maintenance (l'employé qui vient recharger la billetterie),

Matériel externe : Imprimante par exemple (matériel différent de celui sur lequel le système s'exécute),

Autres systèmes : Autres machines sur le réseau.

A.4.2.Composants :

Il existe 4 grands types de composants en UML.

A.4.2.1. Composants structuraux :

Ils sont l'équivalent des noms dans le langage et véhiculent la partie statique du modèle.

On en dénombre 7 dont voici les noms et la représentation graphique :

Classe : Définit un modèle qui s'instancie en un ou plusieurs objets. La classe est représentée graphiquement par un rectangle découpé en trois parties : une pour le nom, une pour les attributs et la dernière pour les services.

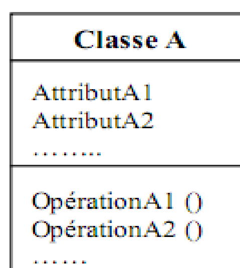


Figure A.1 Classe.

La représentation graphique permet également de faire apparaître la portée (privée indiquée par le caractère -, publique indiquée par +, etc.) des attributs et méthodes. On utilise la même représentation pour des objets, ainsi on dessinera de la même manière la classe Voiture ou l'objet ma Voiture ou bien encore un objet quelconque (anonyme) de cette classe auquel on donnera le nom : Voiture. On distingue les objets par leur nom qui est souligné.

◊ **Interface :** Constituée d'une collection d'opérations qui spécifient un service d'une classe ou d'un composant (complet ou partiel). L'interface est un « contrat » de la spécification de l'ensemble des traitements que la classe s'engage à effectuer si elle veut prétendre disposer d'une certaine interface.

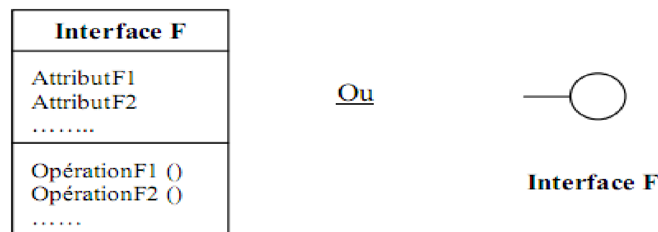


Figure A.2 Interface.

- **Collaboration :** La collaboration permet de regrouper des éléments en interaction. Elle a une dimension structurelle, mais aussi comportementale dans la mesure où elle peut permettre d'englober des enchaînements d'envoi de message. On la représente graphiquement par une éllipse en pointillés.

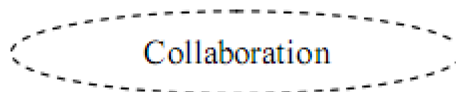


Figure A.3 Collaboration.

- **Classe active :** La classe active est une classe qui a la particularité de posséder un fil d'exécution (un ou plusieurs processus ou threads), elle peut donc initier une activité. On la représente graphiquement de manière identique à la classe, mais avec un contour en gras.

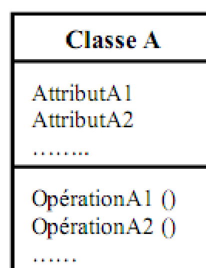


Figure A.4 Classe active

- **Composant:** Le composant est une partie d'un système qui se conforme à une spécification et fournit la réalisation d'un ensemble d'interfaces. Il peut être constitué

de plusieurs sous composants. C'est l'emballage physique de composants logiques (classes, interfaces, collaborations...)

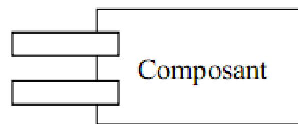


Figure A.5 Composant

- **Nœud:** Le nœud est l'élément physique, la ressource informatique sur lequel des composants peuvent résider ou migrer.

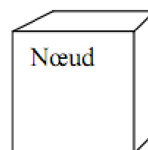


Figure A.6 Nœud

A.4.2.2. Composants comportementaux :

Ils sont l'équivalent des verbes dans le langage et véhiculent la partie dynamique du modèle.

Ils connectent entre eux des éléments structurels tels que collaborations et objets. On distingue deux types :

- **Interaction :** Elle correspond à un envoi de messages entre objets pour accomplir un but spécifique. Une interaction implique des messages, des séquences d'actions (le comportement induit par un message) et des liens (connexions entre objets).

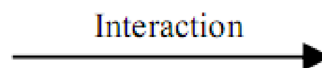


Figure A.7 Interaction.

Il existe toute une variété de messages qui peuvent être :

∗ Asynchrones : l'émetteur n'attend pas que le récepteur ait traité le message,

∗ Synchrones : l'émetteur attend que le récepteur ait traité le message avant de continuer,

Etat: La machine à états spécifie le cycle de vie d'un objet quand cet objet a un comportement dynamique. On représente graphiquement un état par un rectangle aux coins

arrondis (au centre), et on distingue les états de départ (à gauche) et terminaux (à droite), comme décrit dans la figure suivante.

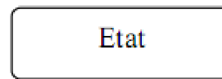


Figure A.8 Etat

A.4.2.3. Composants de regroupement :

Un package est une entité purement conceptuelle destinée à organiser d'autres éléments, qu'ils soient structurels, de comportement ou même d'autres packages.

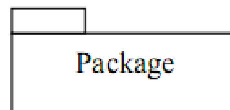


Figure A.9 Package

A.4.2.4. Composants d'annotation :

Ils permettent d'inclure des explications supplémentaires sur un diagramme.

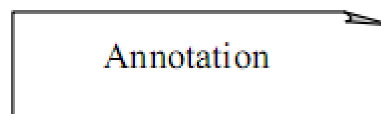


Figure A.10 Annotation

A.4.3. Relations :

Il existe quatre types de relations en UML, chacune symbolisée différemment :

A.4.3.1. Dépendance :

La dépendance est une relation sémantique indiquant que B dépend de A, et donc que si A est modifié, B le sera également. La relation de dépendance possède un nom et se représente graphiquement par :

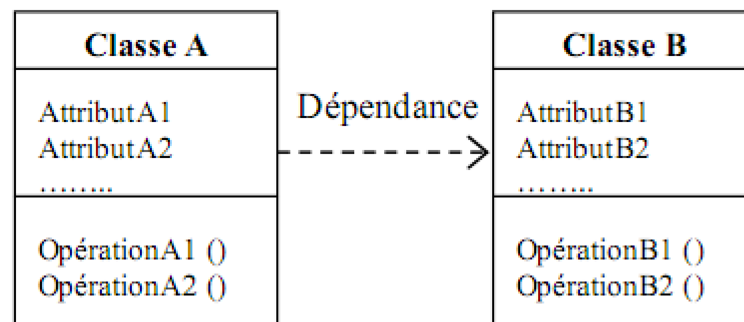


Figure A.11 Dépendance

A.4.3.2. Association :

L'association est une relation structurelle. Elle regroupe l'ensemble des liens qui connectent des objets entre eux. En général, Pour qu'un objet A puisse envoyer un message à un objet B, il faut que A connaisse l'adresse de B, et donc qu'il existe une association entre les deux objets. L'association peut exister entre objets ou entre classes. Elle porte un nom, et les deux extrémités de l'association portent un nom de rôle, le rôle des éléments mis en jeu dans l'association, ainsi qu'une valeur de multiplicité, le nombre des éléments mis en jeu.

On la représente graphiquement de la manière suivante :

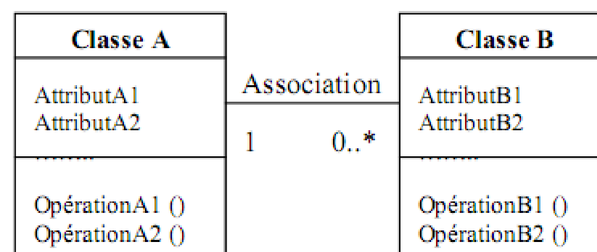


Figure A.12 Association

- **Agrégation :** Cas particulier de relation d'association qui indique qu'une classe est une partie d'une autre classe. Cette relation comporte également des ordres de multiplicité.

On la représente graphiquement en décorant la relation d'association par un losange.

Les agrégations n'ont pas besoin d'être nommées : implicitement elles signifient « contient», « est composé de ».

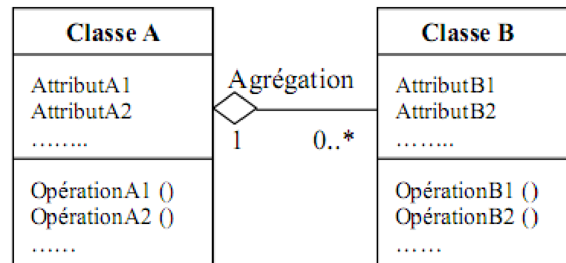


Figure A.13 Agrégation

composition : est une agrégation plus forte impliquant que :

* Un élément ne peut appartenir qu'à un seul agrégat composite.
* La destruction de l'agrégat composite entraîne la destruction de tous ses éléments.

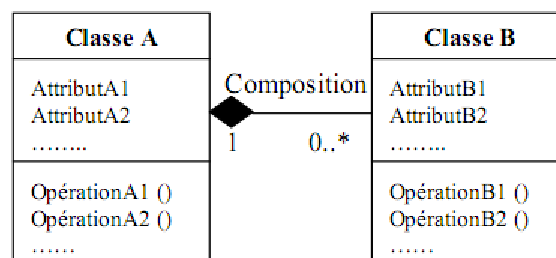


Figure A.14 Composition

A.4.3.3. Généralisation :

La généralisation est une relation structurelle entre classes qui représente l'héritage. On la représente par :

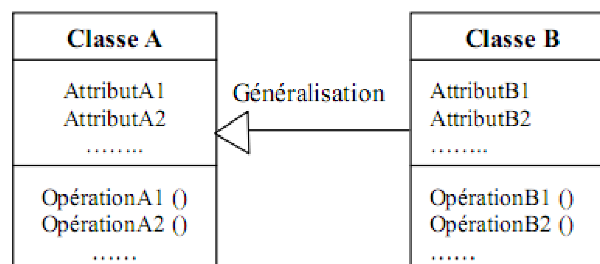


Figure A.15 Généralisation

A.4.3.4. Réalisation :

La réalisation est une relation sémantique utilisée principalement à deux endroits : soit pour indiquer qu'une interface est réalisée par une classe, soit pour indiquer qu'un cas d'utilisation est réalisé par une collaboration d'objets. On la représente par :

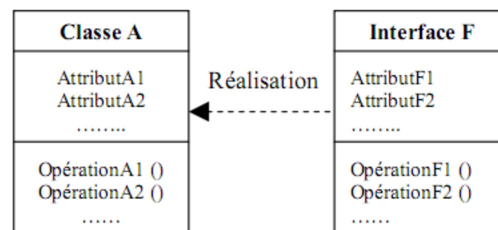


Figure A.16 Réalisation

A.5. Diagrammes en UML :

UML est un langage graphique et repose sur neuf types de diagrammes. Chacun de ces diagrammes utilise le même principe : les concepts sont représentés par des symboles, et les relations entre les concepts sont représentées par des lignes qui relient les symboles.

Le vocabulaire et la grammaire d'UML sont très réduits. [22]

A.5.1. Diagrammes des cas d'utilisation (Use cases) :

Les cas d'utilisation décrivent le comportement du système du point de vue de l'utilisateur. Ils permettent de définir les limites du système et les relations entre le système et son environnement. Un cas d'utilisation est une manière spécifique d'utiliser le système. C'est l'image d'une fonctionnalité déclenchée en réponse à la stimulation d'un acteur externe. Ils permettent de centrer la construction du système sur les besoins des utilisateurs.

Les diagrammes font intervenir les acteurs, les autres systèmes et les cas d'utilisation eux-mêmes.

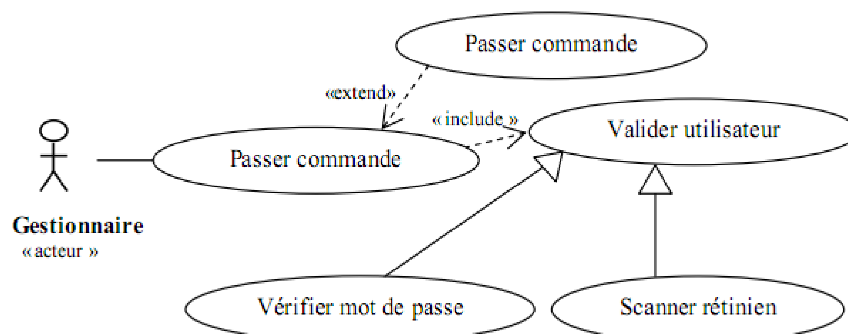


Figure A.17 Exemple d'un diagramme des cas d'utilisation

Relation entre les cas d'utilisation :

En plus de la relation de communication, qui consiste au déclenchement d'un cas d'utilisation par un acteur, nous pouvons citer deux types de liens ou relations qui sont les plus utilisés : le lien d'inclusion et le lien d'extension.

Relation d'inclusion (include): Une relation d'inclusion de cas d'utilisation A par apport à un cas d'utilisation B signifie qu'une instance de A contient le comportement décrit dans B, le cas d'utilisation A ne peut pas être seul.

Relation d'extension (extend) : Une relation d'extension de cas d'utilisation A par apport à un cas d'utilisation B signifie qu'une instance de A peut être étendu par le comportement décrit dans B.

A.5.2. Diagrammes d'objets :

Ces diagrammes permettent une représentation en deux dimensions et statique des relations entre objets. En général, on leur préfère les diagrammes de collaboration, plus informatifs, décrits ci-dessous.

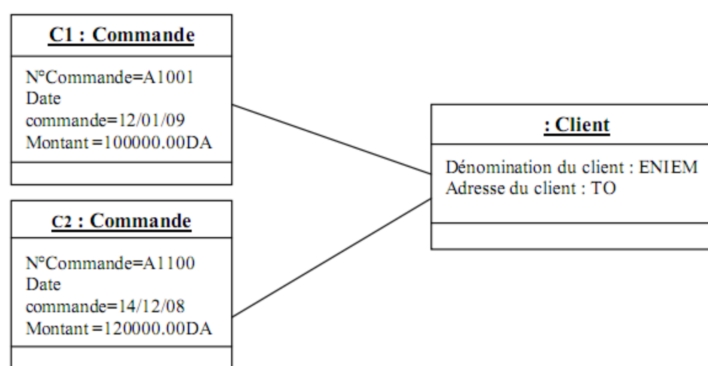


Figure A.18 Exemple d'un diagramme d'objet

A.5.3. Diagrammes de collaboration :

Ce sont des diagrammes similaires aux diagrammes d'objets, mais en plus on y fait figurer les envois de messages, annotés par leur ordre d'apparition.

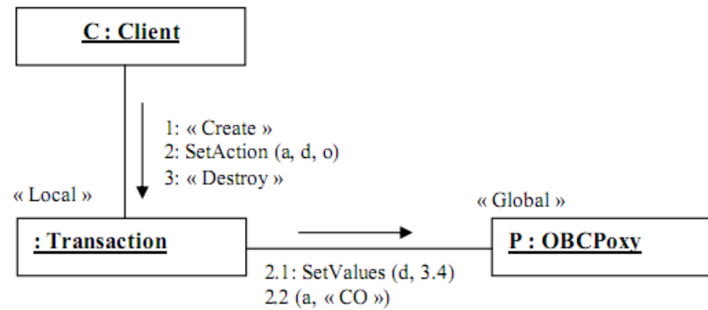


Figure A.19 Exemple d'un diagramme de collaboration

A.5.4. Diagrammes de séquence :

Ces diagrammes représentent également la dynamique de fonctionnement du système. En revanche, la représentation temporelle des événements est mise en avant plutôt que la représentation spatiale.

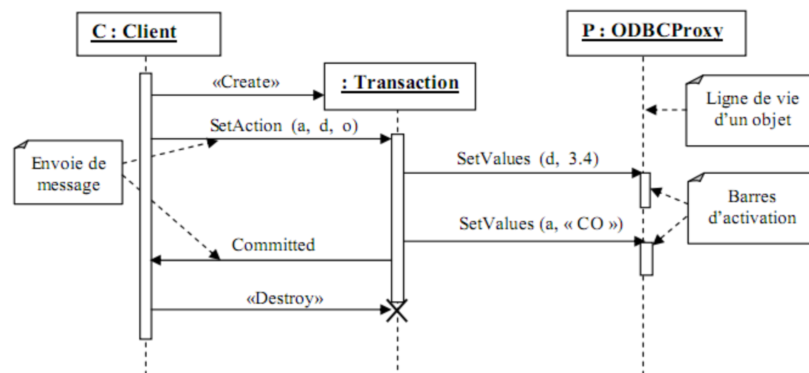


Figure A.20 Exemple d'un diagramme de séquence

A.5.5. Diagrammes de classes :

Ces diagrammes décrivent l'architecture du système; on y représente les classes et les relations entre classes, qu'elles soient d'héritage, d'agrégation, etc.

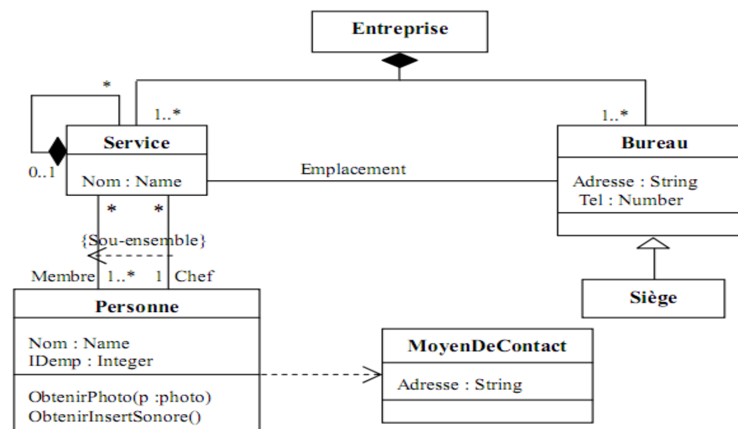


Figure A.21 Exemple d'un diagramme de classes

A.5.6. Diagrammes d'états :

Les diagrammes précédents ne suffisent pas à prendre en compte tous les aspects dynamiques d'un système, ou pas suffisamment simples dans tous les cas. Il est fréquent qu'on ait besoin de modéliser des automates à états finis, qu'il est fastidieux de décrire par des diagrammes de séquence. Dans ce cas, on préfèrera les diagrammes d'états qui ont été introduits spécifiquement pour cet usage.

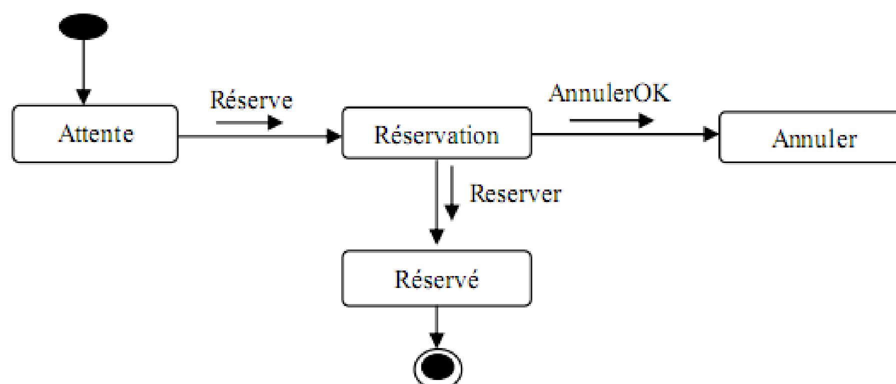


Figure A.22 Exemple d'un diagramme d'états.

A.5.7. Diagrammes d'activités :

Ce sont des variantes des diagrammes d'états, organisés par rapport aux actions et destinés à représenter le comportement interne d'une méthode (réalisation d'une opération) ou d'un cas d'utilisation.

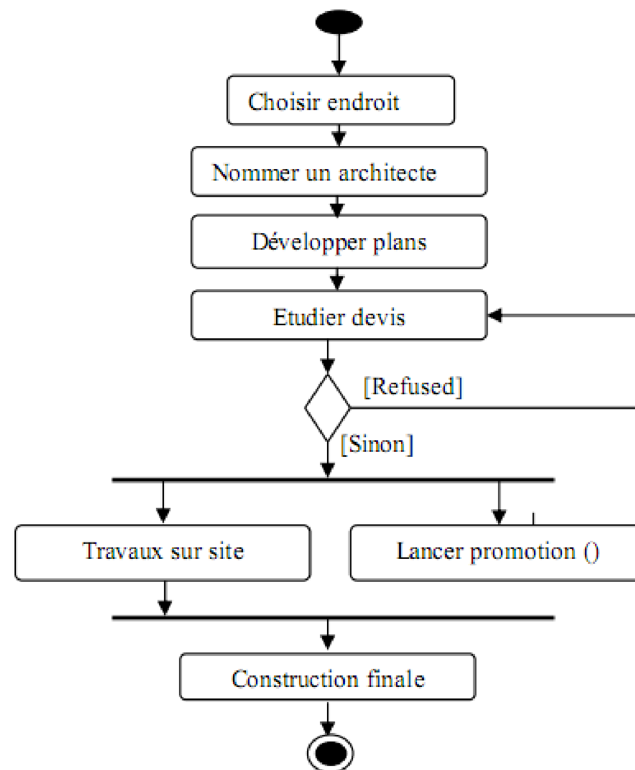


Figure A.23 Exemple d'un diagramme d'activité.

A.5.8. Diagrammes de composants :

Ils définissent les assemblages de fichiers. Ils peuvent servir aussi bien pour les fichiers de documentation, le code source ou les exécutables. Ils sont très utiles pour documenter non seulement le projet, mais également sa gestion: on peut faire apparaître les fichiers UML dans ces diagrammes.

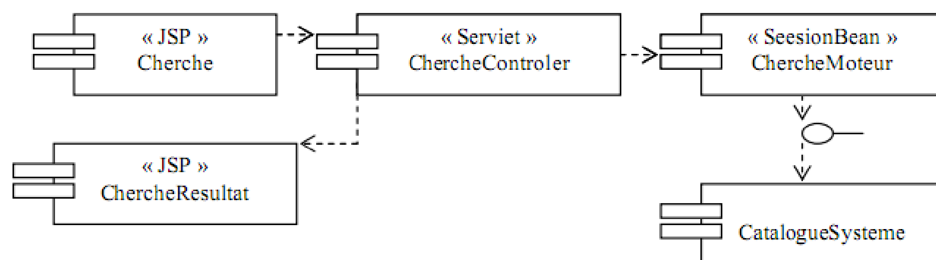


Figure A.24 Exemple d'un Diagramme de composants

A.5.8. Les diagrammes de déploiement :

Ils définissent la répartition des packages sur les nœuds physiques. Ils sont surtout utiles dans le cas de grosses applications réparties.

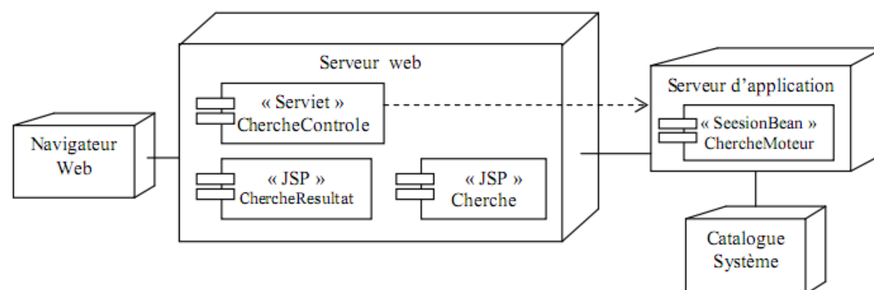


Figure A.25 Exemple d'un Diagramme de déploiement

A.6. UML et les AGL :

De nombreux acteurs du marché proposent des outils qui supportent UML. Le nombre d'ateliers de génie logiciel orientés objet ne cesse de croître.

Rational Software est bien sûr un des acteurs majeurs dans la promotion d'UML. Leur produit Rational Rose est certainement l'un des outils les plus avancés dans le support d'UML.

L'objectif de ces AGL est de prendre en charge au maximum le développement et le suivi du projet : dessin et vérification de cohérence des diagrammes UML, gestion de la documentation, des versions de projet mais également génération automatique du code à partir des diagrammes UML.

A.7. Conclusion :

Le langage unifié, comme son nom l'indique, définit un langage visuel universel et unique qui doit permettre de statuer sur une manière claire de communiquer la façon de faire d'un système. Cet outil a été développé afin de permettre l'avènement d'un processus de développement standardisé. Le langage UML est de ce fait plus expressif, plus propre et plus uniforme que le formalisme des langues utilisées dans les autres méthodes de conception du marché. UML supprime ainsi les notations et terminologies inutiles qui obscurcissaient les similarités de bases des différentes approches.

UML, comme l'on a vu, ne propose pas une démarche objet mais une notation adaptée au monde de développement orienté objet. Il nous a donc permis de s'initier aux techniques de modélisation objet. La notation UML peut s'adapter à tous les projets informatiques.

Annexe B. Extension d’UML pour le Web :

B.1-Définitions :

Une extension d’UML est définie par les nouveaux stéréotypes, étiquettes et contraintes. En les combinant, on peut créer de nouvelles briques de base pouvant être utilisées dans un modèle :

Un stéréotype : est une extension du vocabulaire d’UML, il permet d’associer une nouvelle signification à un élément du modèle. On représente un stéréotype par une chaîne de caractères entre guillemets (**« »**).

Une étiquette : est une extension des propriétés d’un élément, elle permet la description d’une nouvelle propriété d’un élément du modèle. On la représente par une chaîne de caractères entre chevrons (<>).

Une contrainte : est une extension de la sémantique d’UML. Elle édicte une règle que le modèle doit vérifier pour être bien qualifié de « bien formé ». Elle est représentée par une chaîne de caractères entre accolades (**{}**).

B.2-stéréotypes :

B.2.1-classe

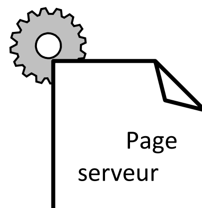
- **Page serveur « server page »**

- **Description :**

Une page serveur représente une page qui possède des scripts exécutés par le serveur. Ces scripts interagissent avec des ressources serveur, telles que les bases de données.

Les opérations de l’objet représentent les fonctions dans le script, et ses attributs représentent les variables qui sont visibles dans la portée de la page.

- **Icône:**



- **Contraintes:**

Les pages serveur ne peuvent avoir de relation qu’avec les objets sur le serveur.

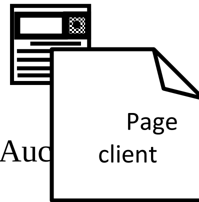
- **Etiquette :** moteur de script pouvant être un langage, ou le moteur qui doit être utilisé pour exécuter ou interpréter cette page.

- **Page client « client page »**

- **Description :**

Une instance d’une page client est une page Web formatée en HTML, Un mélange de données, de représentation et même de logique ; les fonctions d’une page client correspondent aux fonctions des scripts de la page Web, quand à ses attributs, ils correspondent aux variables déclarées dans les scripts et qui sont accessibles à toutes les fonctions de la page. Les pages client peuvent avoir des associations avec d’autres pages client ou serveur.

- **Icône:**



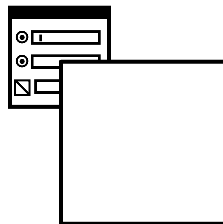
- **Contraintes:** Auc
- **Etiquette :**

Titre : titre de la page tel qu'il est affiché dans le navigateur.
Base: URL de base pour déréférencer les URL relatives
Corps : ensemble des attributs de la balise <body> qui définit les caractéristiques par défaut du texte et de l'arrière-plan.

- **Formulaire « form »**
- **Description :**

Une classe stéréotypée « form » est un ensemble de champs de saisie faisant partie d'une page client. A une classe formulaire correspond une balise HTML <form>. Ses attributs sont les éléments de saisie d'un formulaire HTML telles une zone de saisie, une zone de texte, bouton d'option…
Un formulaire n'a pas d'opérations : toutes les opérations qui interagissent avec le formulaire appartiennent à la page qui les contient.

- **Icône:**



- **Contraintes:**

Aucune

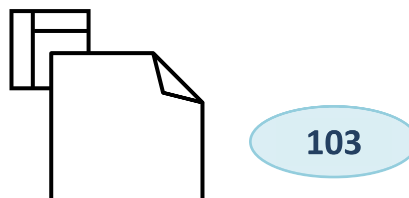
- **Etiquette :**

Les méthodes Post ou Get utilisées pour soumettre les données à l'URL de l'attribut action de la balise HTML <form>

- **Structure de cadres « Frameset »**
- **Description :**

Une structure de cadres est un conteneur de plusieurs pages web. La zone d'affichage rectangulaire est divisée en cadres rectangulaires et à chaque cadre peut être associé un nom unique de cible « target ».
Le contenu d'un cadre peut être une page web ou une structure cadre.
Une classe stéréotypée « frameset » est directement associée à une structure de cadre de page par la balise HTML <frameset>

- **Icône:**



- **Contraintes:**

  Aucune

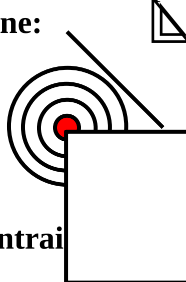
- **Etiquette :**

  Rangées : valeur de l'attribut rows de la balise HTML<frameset>. C'est une chaîne de pourcentages séparés par des virgules. Définissant les hauteurs relatives des cadres.
  Colonnes : valeur de l'attribut de cols de la balise HTML<frameset>.

- **Cible « Target »**
- **Description :**

Une cible est une zone nommée dans la fenêtre de navigation dans laquelle les pages Web peuvent être affichées. Le nom de la classe stéréotypée est celui de la cible. Souvent une cible est le cadre défini dans une fenêtre, mais elle peut être une toute nouvelle instance du navigateur : une nouvelle fenêtre. Une association « targeted link » spécifie la cible où une page Web doit être affichée.

- **Icône:**



- **Contrai**

Un nom de cible doit être unique pour chaque client du système. Par conséquent une seule instance d'une même cible peut exister sur un même client.

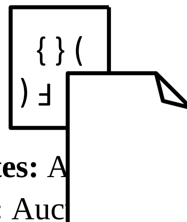
- **Etiquette :**

  Aucune

- **Objet javascript « javascript object »**
- **Description :**

Sur un navigateur compatible javascript, il est possible de simuler des objets personnalisés à l'aide de fonctions javascript. Ces objets ne peuvent exister que dans le contexte de pages client.

- **Icône:**



- **Contraintes:** A
- **Etiquette:** Auc

B.2.2 Association

- **Lien « link »**

- **Description :**

Un lien est un pointeur d'une page client vers une autre page. Dans un diagramme de classe, un lien est une association entre une page client et une page serveur ou une page client. Un lien correspond à une balise HTML.

- **Icône:** Aucune
- **Contraintes:** Aucune

Etiquette : Aucune.

- **Paramètre :** c'est la liste de noms de paramètres qui doivent être passés avec la demande de la page liée.

- **Lien cible « targeted link»**
- **Description :**

Similaire à une association lien, un lien cible est un lien dont la page associée est affichée dans une cible. A un lien correspond une balise HTML ancre, dont l'attribut target prend la valeur de la cible.

- **Icône:**

Aucune

- **Contraintes:**

Aucune

- **Etiquette :**

Nom de la cible (target name): en plus de la liste des paramètres, le nom de la cible ou de la page vers laquelle pointe le lien doit être affiché.

- **Contenu de cadre « frame content »**
- **Description :**

Une association contenue de cadre est une association d'agrégation qui traduit l'apparence d'une page ou de cible à un cadre.
Une association contenue de cadre peut aussi pointer vers une structure de cadre aboutissant dans ce cas à des cadres imbriqués.

- **Icône:**

Aucune

- **Contraintes:**

Aucune

- **Etiquette :**

Rangée : entier qui indique la rangée du cadre dans la structure de cadres auquel appartient la page, ou la cible associée.
Colonne: entier qui indique la colonne du cadre dans la structure des cadres auquel appartient la page, ou la cible associée.

- **Soumet « submit »**

- **Description:** Une association de soumission se trouve toujours entre un formulaire et une page serveur. Les formulaires soumettent les valeurs de leurs champs au serveur pour qu'il les traite. Le serveur Web traite la page serveur, qui accepte les informations du formulaire.
- **Icône:** Aucune
- **Contraintes:** Aucune
- **Etiquette :**

Paramètres : la liste de nom de paramètres qui doivent être passées avec la demande de la page liée.

- **Constuct «build »**
  - **Description :**

La relation « **build** » est une relation particulière qui fait le pont entre les pages clients et les pages serveur.

L'association « **build** » identifie quelle page serveur est responsable de la création d'une page client. C'est une relation orientée, puisque la page client n'a pas connaissance de la page serveur qui l'a construite.

Une page serveur peut construire plusieurs pages client, en revanche une page client ne peut être construite que par une seule page serveur.

- **Icône:** Aucune
- **Contraintes:** Aucune
- **Etiquette:** Aucune

- **Redirige « redirect »**
- **Description :**

Une relation de ce type est une association unidirectionnelle avec une autre page web, pouvant être dirigée à partir d'une page client ou serveur vers une autre page client ou serveur.

Si la relation part d'une page serveur, le traitement de la requête de la page peut se poursuivre par l'autre page. Cela ne veut pas dire que la page de destination participe toujours à la construction de la page client, simplement qu'elle le pourrait, et le déclenchement d'une redirection doit être programmé dans le code de la page d'origine.

- **Icône:** Aucune
- **Contraintes:** Aucune
- **Etiquette**

Délai : délai que doit observer une page client avant de se rediriger à la page de destination. Cette valeur correspond à l'attribut **content** de la balise **<META>.**

- **IIOP « Internet Inter-ORB Protocol »**
  - **Description :**

IIOP est un type spécial de relations entre objets sur le client et d'autres sur le serveur, c'est un mécanisme de communication entre client et serveur différent de HTTP.

- **Icône:** Aucune

- **Contraintes:** Aucune
- **Etiquette :** Aucune

- **RMI « Remote Method Invocation ».**
  - **Description :**

RMI est un mécanisme qui permet à des applets Java, ou à des JavaBeans d'envoyer des messages à des JavaBeans situés sur d'autres machines.

- **Icône:** Aucune
- **Contraintes:** Aucune
- **Etiquette :** Aucune

B.2.3 Attribut :

- **Eléments de saisie « input élément »**
  - **Description :**

Un élément de saisie correspond à la balise <**input**> d'un formulaire HTML ; cet attribut est utilisé pour saisir un mot ou une ligne de texte. Les étiquettes associées à cet attribut stéréotypé, correspondent aux attributs de la balise <**Input**>.

Les attributs obligatoires de la balise HTML <**Input**> sont renseignés de la manière suivante : l'attribut **name** prend la valeur du nom de l'élément de saisie et l'attribut **value** prend celle de sa valeur initiale.

- **Icône:** Aucune
- **Contraintes:** Aucune
- **Etiquette**

Type : le type de l'élément de saisie : texte, numérique, mot de passe, case à cocher, bouton d'option, bouton submit ou bouton reset.

- **Sélection d'élément «select element »**
  - **Description :**

Contrôle de saisie employée dans les formulaires permettant à l'utilisateur de sélectionner une ou plusieurs valeurs dans une liste. La plupart des navigateurs restituent ce contrôle par une liste d'options ou une liste déroulante.

- **Icône:** Aucune
- **Contraintes:** Aucune
- **Etiquette**

Taille: définit le nombre d'éléments qui doivent être affichés simultanément.

Multiple : valeur booléenne qui indique que plusieurs éléments peuvent être sélectionnés conjointement.

- **Zone de texte « text aera element »:**
  - **Description :**

Contrôle de saisie, employée dans les formulaires ; qui permet l'écriture de plusieurs lignes de texte.

- **Icône:** Aucune
- **Contraintes:** Aucune
- **Etiquette**

Lignes : nombre de lignes de texte visibles.
Colonnes : largeur visible du texte en largeur de caractère moyenne.

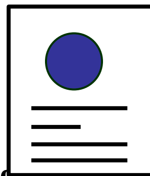
B.2.4 composant :

- **Page web « web page »**
- **Description :**

Un composant page est une page web. Un composant page web peut contenir des scripts client ou serveur.

Souvent le composant page est un fichier texte accessible au serveur web, mais il peut être également un module compilé, chargé et exécuté par le serveur web ; dans les deux cas le serveur web produit à partir du composant page, un document au format HTML, qui est renvoyé en réponse à la requête du navigateur.

- **Icône:**



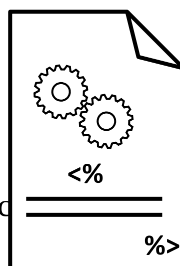
- **Contraintes:** Aucune
- **Etiquette**

Chemin (path) : chemin requis pour spécifier la page web sur le serveur, cette valeur doit être relative au répertoire racine du site de l'application.

- **Page ASP « ASP page »**
- **Description :**

Une page ASP (Active Server Pages) est une page web qui implémente du code ASP coté serveur. Ce stéréotype n'est pertinent que dans un environnement d'applications basé sur les ASP.

- **Icône:**



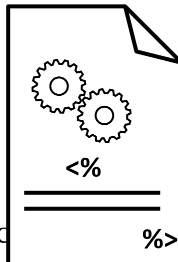
- **Contraintes:** Auc
- **Etiquette :**

Identique à celle de la page web.

- **Page JSP « JSP page »**
  - **Description :**

Une page JSP (Java Server Page) est une page web qui implémente du code JSP coté serveur. Ce stéréotype n'est pertinent que dans un environnement d'applications basé sur les JSP.

- **Icône:**



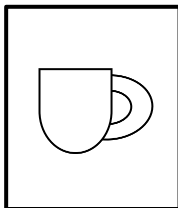
- **Contraintes:** Auc
- **Etiquette**

Identique à celle de la page web.

- **Servlet « servlet »**
  - **Description :**

C'est un composant servlet Java, il n'est pertinent que dans un environnement d'application compatible avec les servlets de Sun.

- **Icône:**



- **Contraintes:** Aucune
- **Etiquette**

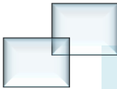
Identique à celle de la page web.

- **Bibliothèque de scripts**
  - **Description :** C'est un composant qui propose un ensemble de sous-routines ou des fonctions pouvant être incluses dans d'autres composants pages web.
  - **Icône:**



« client page » **« server page »** **« frames et »** **« target »** **« form »**

« client « link » « link » « link » «dependac « aggregati



page »	« redirect » « target link »	« redirect » « target link »	« redirect » « target link »	y»	on »
« server page »	« build » « redirect »	« redirect »	« build » « redirect »		
« frameset »	« frame content »		« frame content »	« frame content »	
« target »					
« form »	«Aggregated by»	« submit »			

Annexe C. quelques fiches d'organisme :

Fonds National d'Investissement
Direction de l'Organisation et de l' Informatique (DOI)



Fiche de Renseignements

INFORMATIONS UTILISATEURS

Utilisateur :
Direction :
Département :
Service :
Fonction:
Etage \ Bureau :
Date de la demande : \ \ 2012

INFORMATIONS MATERIELS

DESIGNATION

MARQUE
SERIAL
CB
STATUT

UC

ECRAN

IMPRIMANTE

Fonds National d'Investissement



Direction de l'Organisation et de l' Informatique (DOI)

FICHE D'INTERVENTION

Utilisateur :
Direction :
Département :
Service :
Fonction:
Etage \ Bureau :
MATRICULE :

Date de la demande :
Matériel :
N° de série \ Code Barre :
Panne Signalée :
Panne constatée :
Opérations Effectuées :

Durée de l'intervention :
Remarque :

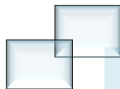
Signature de l'Utilisateur **Signature de l'intervenant**

Rapport d'interventions de l'année 2011

Le nombre d'interventions entre le 02/01/2011 et le 31/12/2011 est de 484 interventions Réparties comme suit :

MOIS	Nombre d'intervention
JANVIER	**70**
FEVRIER	**29**
MARS	**30**
AVRIL	**29**
MAI	**33**
JUIN	**41**
JUILLET	**26**
AOUT	**42**
SEPTEMBRE	**34**
OCTOBRE	**39**
NOVEMBRE	**55**
DECEMBRE	**56**
TOTAL	**484**

DIR \ MOIS	Ja nvier	Fé vrier	M ars	A vril	M ai	uin	J uillet	A out	Septe mbre	Oct obre	Nove mbre	Déce mbre	T OTAL
DFC	5	8	5	5	7		4	5	6	3	7	5	6 7



DED	1 0	4	3	3	3		2	5	4	6	6	11	6 2
DEP	1 4	5	7	7	9	2	5	1 7	10	9	14	10	1 18
DPM	6	4	9	7	7	0	7	9	10	7	10	11	9 7
DCRE	2 1	4	4	2	2		0	3	3	2	6	9	6 1
DOI	3	3	0	4	3		2	0	2	4	2	0	2 5
DRC	2	1	2	0	0		1	0	2	3	5	4	2 0
DG	9	0	0	0	0		3	3	0	5	5	6	34

Interventions par direction :

Direction	DFC	DED	DEP	DPM	DCRE	DOI	DRC	DG	Total
Interventions	67	62	118	97	61	25	20	34	484

Types d'intervention :

Type	Imprimante	Ecran	UC	Réseau	Software	Autre	Total
Nombre	148	34	124	67	83	28	484